République Algérienne Démocratique et Populaire
Ministère de l'enseignement supérieur et de la recherche scientifique

Université Saad Dahlab blida 1
Faculté des sciences
Département d'Informatique

Mémoire de fin d'études
pour l'option du diplôme de master
Option : Sécurité de système d'information

---

un modèle d'apprentissage en profondeur pour détecter les malwares

---

Réalisé par :
Berriche fouad
Feddak Fayçal Abdelghani

Encadré par :Mme. Boustia

septembre 2022

Remerciement

Tout d'abord, nous remercions Dieu le Tout-Puissant, de nous avoir donné la volonté et le courage de réaliser ce travail.

Un grand merci à notre promotrice, Mme N.Boustia qui nous a beaucoup aidé dans la correction et la rédaction de ce document, et aussi pour sa patience, sa disponibilité, et pour avoir accepté de mener ce travail.

Nous remercions sincèrement les membres du Jury de nous avoir fait l'honneur d'accepter et d'évaluer notre travail et tout le corps professoral du département d'informatique de l'Université Saad Dahlab Blida1 pour les efforts fournis dans notre formation. Enfin, nous voudrions exprimer notre profonde gratitude et nos vrais sentiments à nos familles, qui nous ont toujours soutenus.

Dédicace Feycal

Je dédie ce modeste travail :

À Mes parents, pour leur amour infini, leurs sacrifices, leur soutien etleurs encouragements,

À Ma chère grand-mère maternelle et à la mémoire de mon grandpère, que Dieu l'accueille dans son vaste paradis,

À la mémoire de mes chers Grands-parents paternels, que Dieu lesaccueille dans son vaste paradis,

À mon cher frère et ma chère sœur,
À tous les membres de ma famille ainsi que mes amis,

À tous ceux qui me sont chers et à toutes les personnes qui m'ont aidéà atteindre ce niveau

Faycal

Dédicace Fouad

Je dédie ce modeste travail :

À mes chers parents, pour tous leurs sacrifices, leur amour, leur tendresse, leur soutien et leurs prières tout au long de mes études

À toute ma famille pour leur soutien tout au long de mon parcours universitaire, Que ce travail soit l'accomplissement de vos vœux, et le fruit de votre soutien infaillible,
Merci d'être toujours là pour moi.

Fouad

Résumé

Les systèmes de détection de logiciels malveillants ont fait l'objet de nombreuses recherches et jouent un rôle important dans la cybersécurité. L'objectif de cette étude est de modéliser un tel système pour aider les administrateurs système et les utilisateurs à détecter et identifier toute violation de sécurité dans leur organisation afin de les prévenir avant de causer des dommages. Dans ce but, nous avons étudié les performances des méthodes de machine learning (ML) appliquées à la détection des malwares pour la cybersécurité. Ensuite, nous avons appliqué une technique de détection basée sur une approche d'apprentissage en profondeur, un réseau de neurones convolutifs (CNN) pour détecter les logiciels malveillants dans les connexions réseau.

Ensuite, nous avons utilisé diverses mesures appliquées pour évaluer les performances de l'apprentissage automatique (précision, rappel, score F1) et deux autres indicateurs de performance importants pour la détection des logiciels malveillants (taux de détection, taux de fausses alarmes)

Mot clé : Cybersécurité, Système de détection des malwares, L’apprentissage profond, L’apprentissage automatique, Machine learning, Deep learing

Abstract

Malware detection systems have been the subject of much research and play an important role in cybersecurity. The objective of this study is to model such a system to help system administrators and users detect and identify any security violations in their organization in order to prevent them before causing damage. For this purpose, we studied the performance of machine learning (ML) methods applied to malware detection for cybersecurity. Then, we applied a detection technique based on a deep learning approach, a convolutional neural network (CNN) to detect malware in network connections.

Next, we used various metrics applied to evaluate machine learning performance (accuracy, recall, F1 score), and two other important performance indicators for malware Detection (detection rate, false alarm rate)

Keyword : Cybersecurity, Malware detection system, Deep learning, Machine learning.

ملخص

كانت أنظمة الكشف عن البرامج الضارة موضوعًا للكثير من الأبحاث وتلعب دورًا مهمًا في الأمن السيبراني. الهدف من هذه الدراسة هو نمذجة مثل هذا النظام لمساعدة مسؤولي النظام والمستخدمين على اكتشاف وتحديد أي انتهاكات أمنية في مؤسستهم من أجل منعها قبل التسبب في ضرر. لهذا الغرض ، درسنا أداء أساليب التعلم الآلي (ML) المطبقة على اكتشاف البرامج الضارة للأمن السيبراني. بعد ذلك ، طبقنا تقنية اكتشاف تعتمد على نهج التعلم العميق ، وهي شبكة عصبية تلافيفية (CNN) لاكتشاف البرامج الضارة في اتصالات الشبكة.

بعد ذلك ، استخدمنا مقاييس مختلفة مطبقة لتقييم أداء التعلم الآلي (الدقة ، الاسترجاع ، درجة F1) ، ومؤشران مهمان آخران للأداء للكشف عن البرامج الضارة (معدل الاكتشاف ، معدل الإنذار الخاطئ)

كلمات مفتاحية : الأمن السيبراني, نظام الكشف عن البرامج الضارة, التعلم العميق,التعلم الالي .

Table des matières

Table des figures

liste des tableaux

Introduction générale

Le développement rapide des technologies de l'information et de la communication (TIC) a conduit les concepteurs d'applications à commercialiser un nombre impressionnant d'applications. La concurrence sur ce marché a atteint son paroxysme. Certains opérateurs n'ont pas exclu d'utiliser des méthodes qui ne sont pas tout à fait légales pour gagner des parts de marché. Leurs méthodes sont souvent basées sur l'espionnage, l'intrusion, la distorsion, etc. Pour ce faire, ils créent souvent des applications avec du code malveillant et présentent leurs produits pour une utilisation dans le cadre de campagnes publicitaires ou, dans des cas extrêmes, gratuitement. Les utilisateurs inexpérimentés sont souvent victimes de ces opérations. La recherche pour corriger ces pratiques est aujourd'hui très active. C'est la cybersécurité.

La plupart des travaux dans ce domaine sont complexes et les phénomènes peuvent être contenus efficacement. Mais les agents malveillants développent également des techniques de plus en plus sophistiquées pour maintenir leur intention malveillante.

Chapitre 1
Malwares

1.1) Introduction :

Les logiciels malveillants sont devenus l'un des cyber risques modernes les plus importants, posant une menace sérieuse pour nos ordinateurs en raison du développement rapide d'Internet. Les statistiques les plus récentes sur la découverte de ce dernier sont horrifiantes. Les systèmes d'information ou les réseaux sont en danger tout le temps en raison du nombre énorme de virus. Ce chapitre fournit une explication complète des logiciels malveillants, une liste des variétés actuelles et leurs familles, ainsi que la façon dont chacun d'eux fonctionne et à quelle profondeur ils peuvent pénétrer un système.

1.2) Définition d'un malware :

Malware est le nom de certaines versions de logiciels malveillants et cruels, notamment le vol d'informations et les logiciels espions. Les logiciels malveillants consistent généralement en un code développé par des pirates pour causer des dommages extrêmes aux données et aux systèmes ou pour obtenir un accès non autorisé à un réseau. Les programmes ou applications indésirables sont généralement livrés par e-mail sous forme de lien ou de fichier et demandent/exigent que l'utilisateur clique sur le lien ou ouvre le fichier pour exécuter le logiciel malveillant [1].

Les logiciels malveillants constituent en fait une menace pour les individus et les organisations depuis l'apparition du virus Creeper au début des années 1970. Le monde a été attaqué par des centaines de milliers de versions différentes de logiciels malveillants, tous conçus pour causer autant de dégâts et de destruction que possible [1].

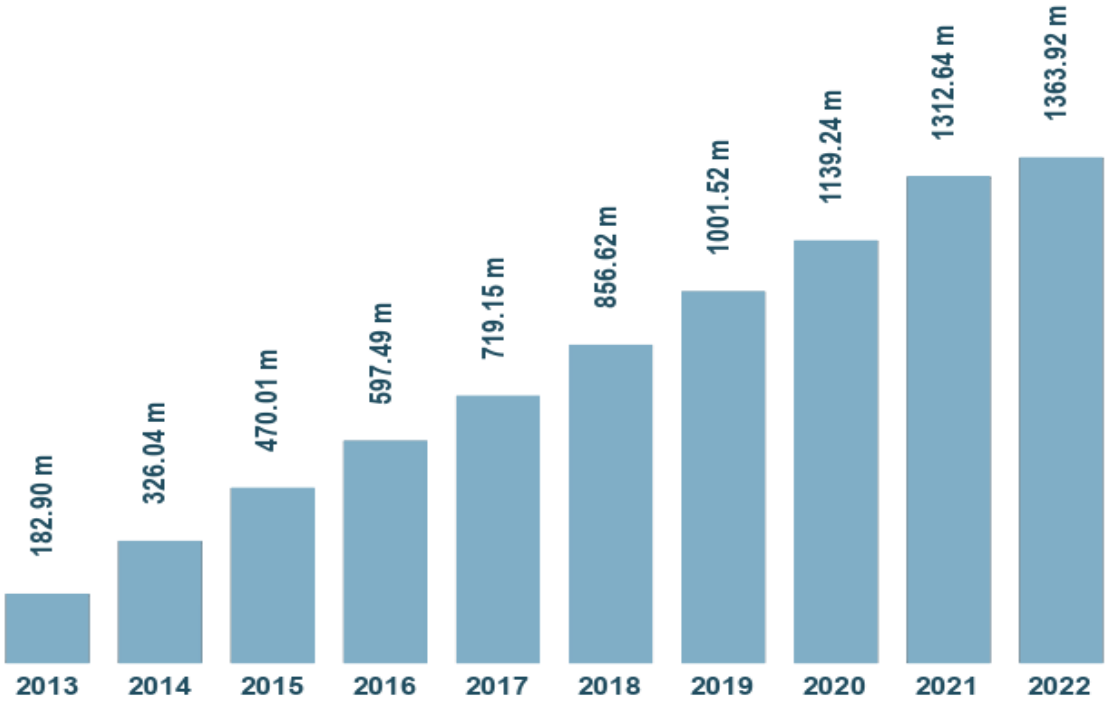


Figure 1.1 : Statistique des détections des malwares 2010-2021 [2]

Figure 1.1 représente l'évolution du nombre des malwares au cours des différentes années, car nous notons que le nombre de programmes augmente chaque année par rapport à l'année précédente, et en 2022, il a atteint un nombre de 1363,92 millions de malwares .

1.3) Classification des logiciels malveillants :

Bien qu'il existe de nombreux types de logiciels malveillants différents, nous pouvons les classer en fonction de leurs caractéristiques et caractéristiques uniques, nous pouvons les décomposer comme suit [3] :

1.3.1) Ver informatique (warm) :

Les vers se propagent via les faiblesses du réseau (qui pourraient être utilisées pour blesser quelque chose ou quelqu'un) ou des attaques de phishing. Une fois qu'un ver s'est installé dans la mémoire de votre ordinateur, il commence à infecter toute la machine et parfois... tout votre réseau Et la figure suivante nous explique les étapes du travail du ver [4].

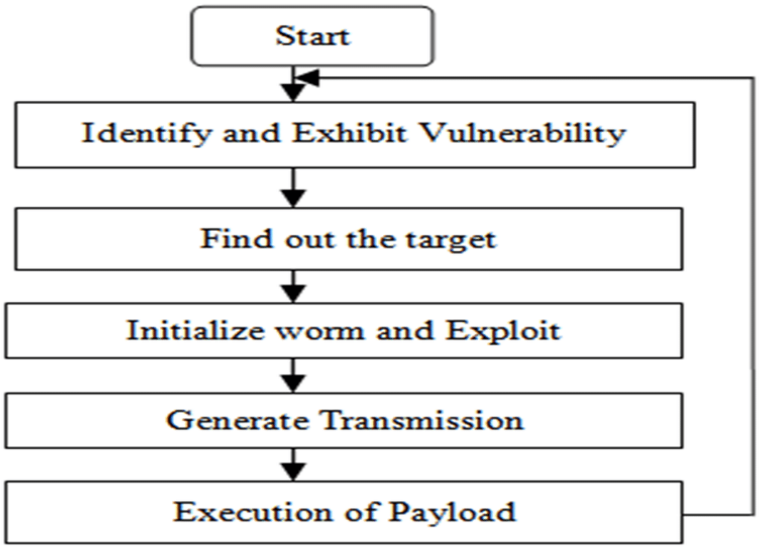


Figure 1.2 Aperçu de la progression du warm [4]

figure 1.2représente le fonctionnement de warm depuis sa execution

Car après sa fabrication et lors de son activation, il passe par 4 étapes :

- Recherche de cible : Lorsqu'un ver pénètre dans un réseau, la première chose qu'il fait est de rechercher une victime à propager et à exploiter. L'analyse de cible aveugle, l'analyse de liste de résultats, l'analyse topologique, la recherche passive et la recherche sur le Web sont quelques-unes des nombreuses approches de recherche de cible. De nombreux vers modernes sont conçus pour attaquer les serveurs Web [4].

- Format de diffusion : Une fois que le ver initial a identifié la cible, il utilise de nombreuses stratégies pour distribuer des copies de lui-même à de nouvelles victimes. Les techniques autoportées, de deuxième canal, intégrées et Botnet sont des exemples de formats de propagation [4].

- Modes de transmission : La transmission des vers est effectuée par des vers TCP et UDP. Les vers TCP ont une latence limitée et sont orientés connexion. Ces vers empêchent le fil d'avancer. Les vers UDP se propagent par autopartage. Ils n'ont pas de connexion et ont une bande passante limitée. Les vers UDP obstruent les ressources du réseau [4].

- Schémas de charge utile : Le code du ver fait référence aux charges utiles. Les vers codés sont plus difficiles à détecter dans un réseau. Les formes de charge utile de

vers incluent les schémas de vers monomorphes, polymorphes et métamorphiques. Sur la base des paramètres susmentionnés, les vers Internet choisissent des cibles, effectuent des copies et déterminent les supports de transmission et les méthodes de charge utile pour une propagation sûre. Les algorithmes de détection antivirus seront modifiés à l'avenir pour mieux détecter les virus en fonction de leurs caractéristiques de propagation [4].

1.3.2) Viruses :

Un virus reste inactif jusqu'à ce qu'il soit activé par le fichier ou le programme hôte infecté, ce qui se fait généralement via des sites Web malveillants, le partage de fichiers ou le téléchargement de pièces jointes. Après cela, le virus a la capacité de se multiplier et de se propager dans tout votre système [5].

Lorsqu'un virus se fixe avec succès sur un logiciel, un fichier ou un document, il reste inactif jusqu'à ce que l'ordinateur ou l'appareil soit obligé d'exécuter son code. Pour infecter votre ordinateur avec un virus, vous devez d'abord démarrer le logiciel infecté par le virus, qui exécute ensuite le code viral [5].

Cela signifie qu'un virus peut rester inactif sur votre ordinateur pendant une longue période sans créer de symptômes. Une fois que votre ordinateur a été infecté par un virus, celui-ci peut se propager à d'autres ordinateurs du même réseau. Un virus peut voler des mots de passe et des données, suivre les frappes au clavier, supprimer des fichiers, spammer vos contacts de messagerie et même prendre le contrôle de votre machine [5].

Alors que certains virus sont conçus pour être divertissants, d'autres peuvent avoir des effets néfastes de grande envergure. Les données peuvent être effacées ou votre disque dur peut être endommagé de façon permanente à la suite de cela. Pire encore, certaines maladies sont conçues dans un but lucratif [5].

1.3.3) Bots et Botnets :

Un bot est un ordinateur qui a été infecté par un logiciel malveillant et qui peut être contrôlé à distance par un pirate. Ce bot (également connu sous le nom d'ordinateur

zombie) peut ensuite être utilisé pour lancer plus d'assauts ou rejoindre un botnet (alias un botnet). Le schéma suivant illustre le fonctionnement des bots et des botnets [6]:

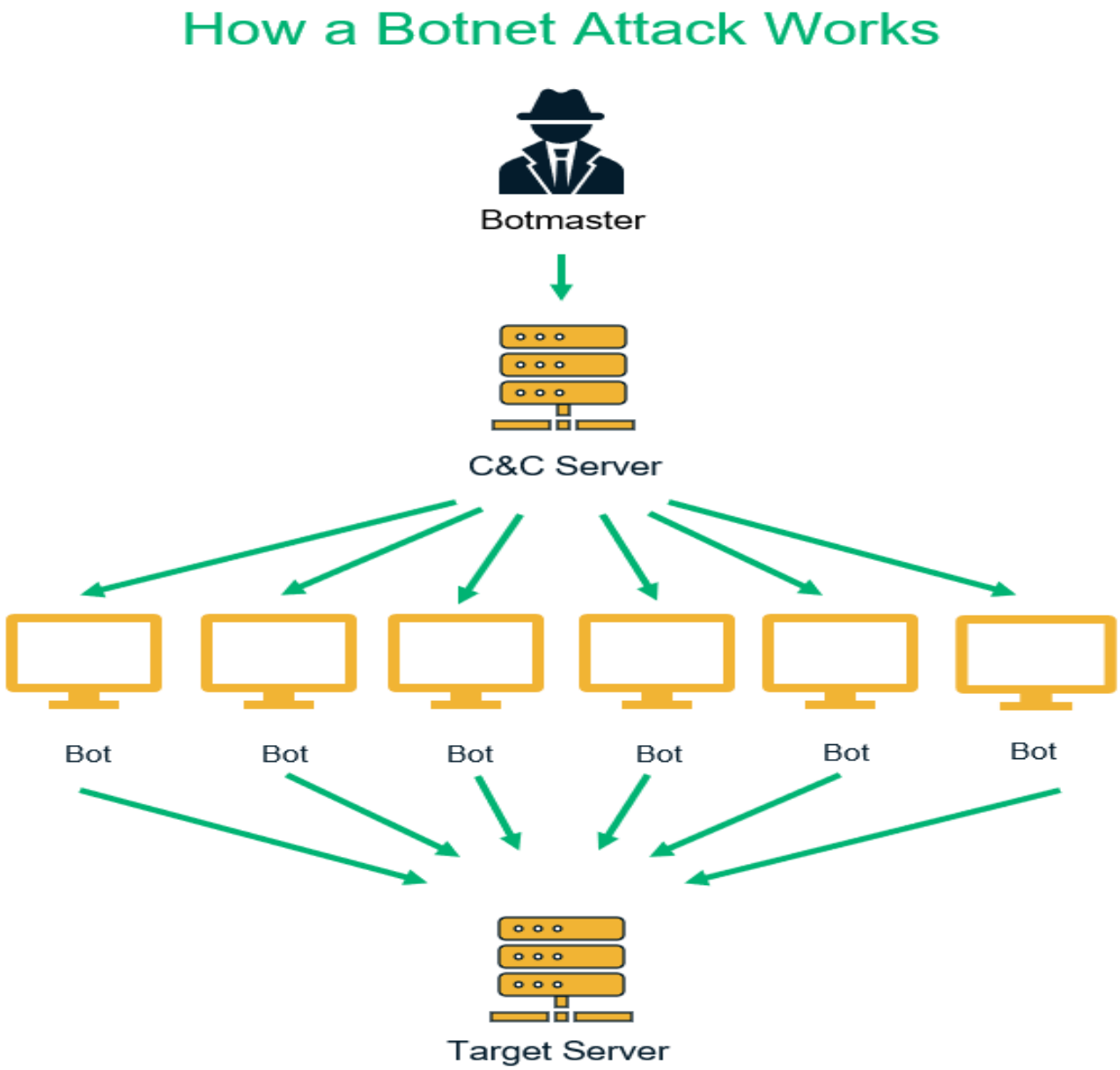


Figure 1.3 : Comment botnet travail [6]

L'attaque du botnet est divisée en trois étapes :

- Phase 1 : Recruter de nouveaux hôtes pour rejoindre l'armée des botnets:

Un assaut de botnet implique un grand nombre de machines infectées liées au même serveur de commande et de contrôle. Un botmaster infecte les nouveaux appareils en utilisant les techniques d'insertion de logiciels malveillants suivantes :

E-mails d'hameçonnage (Phishing emails) : L'attaquant envoie des e-mails de phishing à ses cibles tout en se faisant passer pour des entreprises respectables, des recruteurs, des équipes de support technique, des employeurs et des collègues. Ces e-mails peuvent inclure des pièces jointes, des macros ou des liens nuisibles qui envoient les destinataires vers un site Web contenant du spam. Le malware botnet peut s'installer automatiquement sur les ordinateurs des utilisateurs lorsqu'ils ouvrent des pièces jointes infectées ou cliquent sur des liens malveillants dans des e-mails [6].

Les sites Web malveillants (Malicious websites) : Les logiciels malveillants peuvent être cachés dans des photographies, des films, des chansons, des diaporamas, des fichiers, des logiciels et des publicités sur certains sites Web. Les logiciels malveillants peuvent également être trouvés via des liens et des boutons. Le cheval de Troie botnet infecte les ordinateurs ou les appareils des utilisateurs lorsqu'ils visitent ces sites et téléchargent des fichiers multimédias infectés ou cliquent sur des liens malformés [6].

Exploits de vulnérabilité (Vulnerability exploits) : Le botmaster recherche les appareils connectés présentant des vulnérabilités connues telles que CVE-2019-3396 et CVE-2020-5902. Ils infectent les appareils avec des logiciels malveillants en exploitant ces failles. Lorsqu'un logiciel malveillant de botnet infecte un appareil, il recherche d'autres appareils vulnérables à l'infection et les force à rejoindre le réseau de botnet [6].

-Phase 2 : établissement de la communication entre le périphérique hôte et le Bot Herder :

Le botmaster crée l'une de ces deux voies pour établir une communication interne dans un botnet.

Réseau client-serveur (Client-Server Network) : Un serveur central unique envoie des commandes à tous les hôtes infectés (clients), qui lui font ensuite rapport. Dans certains cas, l'attaquant communiquera en utilisant le réseau Internet Relay Chat (IRC). Tous les clients piratés doivent connaître le bon serveur, port et canal IRC auquel se connecter afin d'interagir avec le serveur maître [6].

Réseau de Botnet Peer-to-Peer (Peer-to-Peer Botnet Network) :

Il s'agit d'un canal de niveau supérieur. Les bots ne reçoivent pas de commandes via un serveur C&C centralisé ; au lieu de cela, ils transfèrent les commandes directement à d'autres bots. Étant donné que les réseaux P2P sont généralement décentralisés, le gouvernement et les chercheurs en cybersécurité sont confrontés à de nombreux obstacles pour les détecter. C'est difficile car les bots ne communiquent pas avec un serveur C&C centralisé, ils ne peuvent donc pas cibler, surveiller ou supprimer un serveur en particulier [6].

-Phase 3 : Utilisation de Botnet Malware pour les cyberattaques :

Lorsqu'un grand nombre d'appareils infectés sont rassemblés sous un parapluie de botnet, les auteurs peuvent les utiliser pour mener une variété d'attaques de botnet [6].

1.3.4) Chevaux de Troie (Trojan Horses) :

Tout logiciel qui trompe les gens sur son véritable objectif est appelé cheval de Troie. Le schéma suivant montre comment fonctionnent les chevaux de Troie :

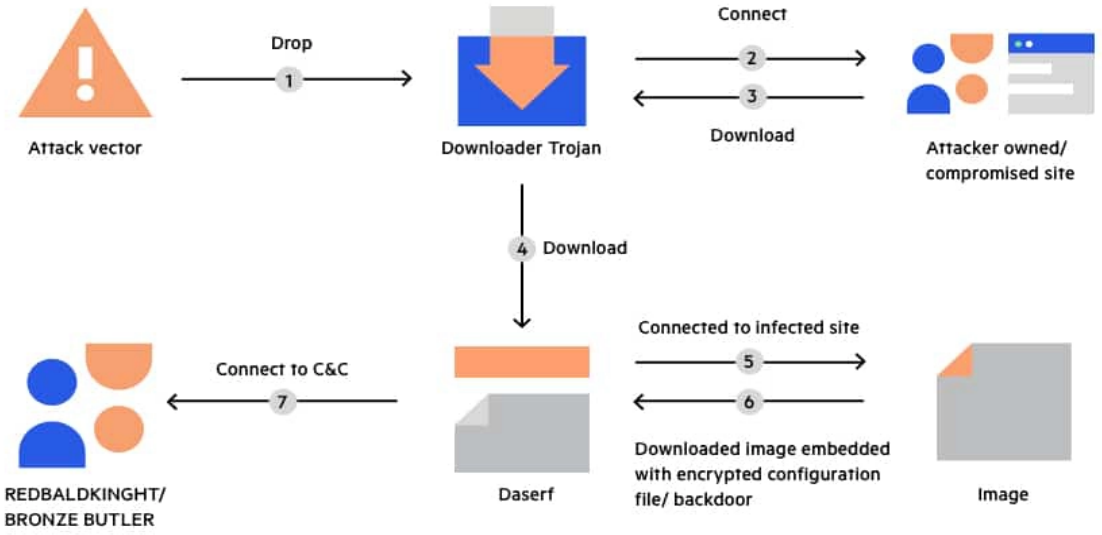


Figure 1.4 : Comment fonctionne le cheval de Troie [7]

Les chevaux de Troie se déguisent en fichiers légitimes afin d'inciter les victimes à les ouvrir, à cliquer ou à les installer. Une fois que cela se produit, le cheval de Troie commence à installer des logiciels, à vous espionner ou à créer d'autres formes de dommages sur votre appareil [8].

Les chevaux de Troie de messagerie, par exemple, imiteront les pièces jointes courantes à l'aide de techniques d'ingénierie sociale. L'e-mail lui-même semblera légitime, mais il s'agit en fait d'une escroquerie par hameçonnage envoyée par un cybercriminel. Le cheval de Troie est activé et commence à attaquer votre appareil dès que vous ouvrez la pièce jointe. La définition du cheval de Troie inclut la tromperie comme élément clé [8].

1.3.5) Logiciels de rançon (Ransomware) :

Un rançongiciel est un logiciel malveillant qui crypte les données et empêche la cible d'y accéder jusqu'à ce qu'une rançon soit payée. Tant que la victime ne paie pas, elle est considérée comme partiellement ou totalement incapable de travailler. Le diagramme ci-dessous illustre le fonctionnement des rançongiciels [9].

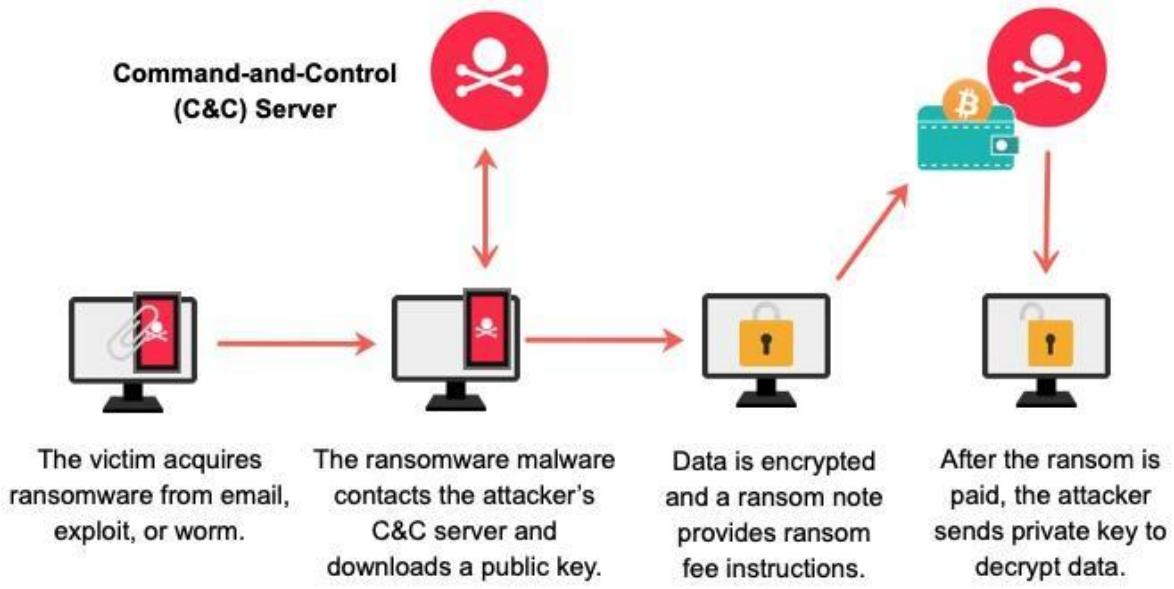


Figure 1.5 : Comment fonctionne un ransomware [9]

Une attaque de ransomware peut être lancée via un lien malveillant, une pièce jointe à un e-mail, une vulnérabilité exploitée, une campagne d'attaque ou un ver. Lorsque le virus rançongiciel infecte l'ordinateur d'une victime, il se propage fréquemment à d'autres appareils du réseau et se connecte à un serveur de commande et de contrôle (C&C) contrôlé par l'attaquant. Le ransomware attend alors une commande de l'attaquant (telle que "chiffrer les fichiers") [9].

Les ransomwares cryptent généralement les fichiers avec un cryptage asymétrique, un processus cryptographique puissant qui crypte et décrypte les données à l'aide de deux clés (une clé privée et une clé publique). L'attaquant contrôle une clé privée et fournit une clé publique au PC de la victime. Le logiciel malveillant commence à chiffrer les données avec les informations de la clé publique. Les ransomwares cryptent généralement les données non critiques en fonction des extensions de fichiers [9].

Le processus de cryptage peut rapidement s'étendre à l'ensemble du réseau et des partages de fichiers une fois que le cryptage des données a commencé. La seule façon de déchiffrer les données est de payer une rançon et d'obtenir la clé privée de l'attaquant [9].

1.3.6) Logiciels publicitaires et escroqueries (Adware & Scams) :

Les logiciels publicitaires sont l'un des types de logiciels malveillants les plus connus. Il affiche des publicités et des pop-ups qui ne vous concernent pas souvent. Les logiciels publicitaires et les escroqueries sont illustrés dans le diagramme ci-dessous .

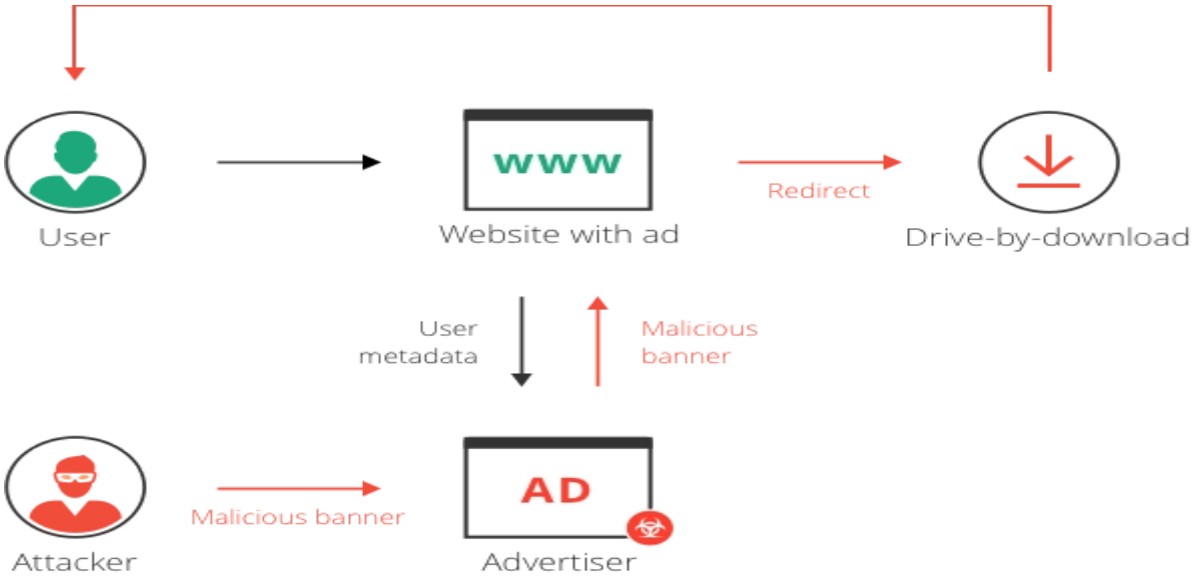


Figure 1.6 : Comment fonctionne Adware & Scams [10]

Les utilisateurs peuvent obtenir un logiciel publicitaire de deux manières : en le téléchargeant accidentellement ou en le récupérant sur un site Web malveillant [10].

Les logiciels publicitaires, contrairement à d'autres types de logiciels malveillants, sont fréquemment téléchargés par l'utilisateur via un autre programme. En réalité, de nombreuses personnes le téléchargent sans le savoir parce qu'elles n'ont pas lu les termes et conditions - alors elles cochent une case et obtiennent le logiciel qu'elles voulaient plus un nouveau programme qu'elles ne veulent pas qui les harcèle avec des publicités [10].

Les logiciels publicitaires peuvent être connectés à des logiciels gratuits distribués via des systèmes de partage de fichiers peer-to-peer, qui sont ensuite installés et s'enfouissent dans l'appareil aux côtés du programme légal. De même, certains développeurs sont connus pour incorporer des logiciels publicitaires dans leurs projets

open source comme moyen de générer des revenus, malgré le fait qu'ils ignorent à quel point cela peut être nocif pour les consommateurs [10].

Certains sites Web, en revanche, vous poussent à télécharger des logiciels publicitaires dès que vous accédez au site ou cliquez sur une page. À moins qu'un antivirus ou un logiciel de blocage des logiciels publicitaires ne soit en cours d'exécution, ceux-ci se téléchargent en arrière-plan et sont extrêmement difficiles à détecter [10].

Après avoir été installé, le logiciel publicitaire commencera à suivre vos activités en ligne et à afficher des publicités, souvent sous la forme d'une multitude de fenêtres contextuelles. Dès son installation, il peut engendrer des problèmes de navigation sur Internet [10].

1.3.6) Les logiciels espions (Spyware) :

Les logiciels espions sont un danger répandu qui est généralement fourni en tant que logiciel gratuit ou partagiciel et possède une fonction frontale attrayante avec un objectif secret fonctionnant en arrière-plan que vous ne détectez peut-être jamais. Il est fréquemment utilisé dans les stratagèmes d'usurpation d'identité et de fraude par carte de crédit [11].

Les logiciels espions peuvent pénétrer dans votre ordinateur par diverses méthodes, y compris les fenêtres contextuelles nuisibles, les téléchargements de logiciels, les pièces jointes aux e-mails et les films et musiques piratés [11].

Une fois qu'un logiciel espion a réussi à accéder à votre ordinateur, il peut effectuer diverses tâches, notamment [11]:

- Exécuter une application qui génère beaucoup de publicités pop-up, ce qui peut rendre votre navigateur moins utilisable.
- Vos recherches sur Internet seront redirigées comme bon vous semble, rendant ainsi les moteurs de recherche inefficaces.
- Les clics, les recherches et, dans les cas extrêmement malveillants, les connexions au compte et les détails de la carte de crédit sont tous enregistrés.
- Modification des paramètres de votre pare-feu pour permettre à davantage de logiciels malveillants d'être reconnus et les tentatives de suppression pour les bloquer.

1.3.7) Hameçonnage (Phishing) :

L'hameçonnage est une sorte d'attaque d'ingénierie sociale, pas un programme malveillant. Cependant, il s'agit d'une stratégie de cyberattaque courante. L'hameçonnage réussit car les e-mails, SMS et liens Web fournis semblent provenir de sources fiables. Le schéma suivant illustre le fonctionnement du spam et du phishing [12]:

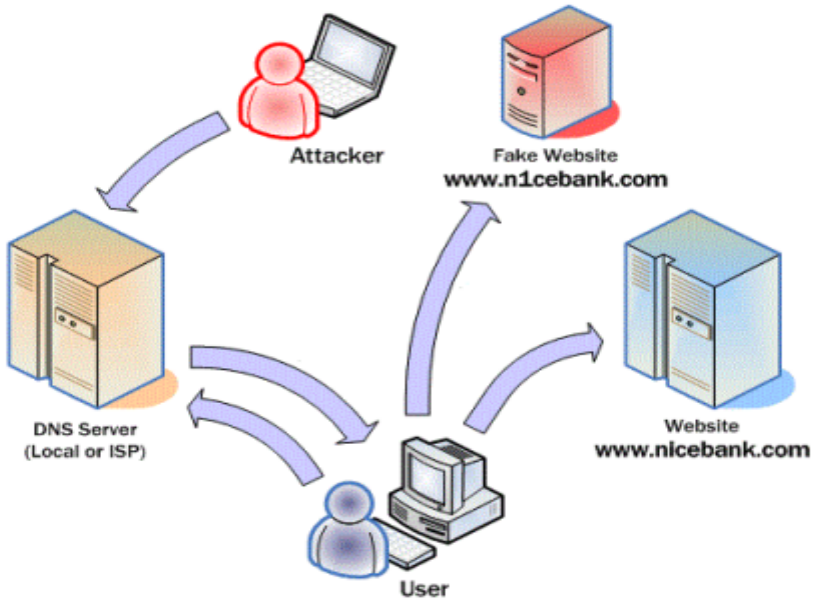


Figure 1.7 : Comment fonctionne le phishing [13]

L'une des deux stratégies principales est utilisée dans la plupart des campagnes de phishing :

Pièces jointes malveillantes : Lorsque des pièces jointes malveillantes, telles que "FACTURE", sont ouvertes, elles installent des logiciels malveillants sur les systèmes des victimes.

Liens vers des sites Web malveillants : Les URL malveillantes dirigent les utilisateurs vers des clones de sites Web légitimes qui téléchargent des logiciels malveillants ou ont des scripts de collecte d'informations d'identification sur leurs pages de connexion.

1.4) Que font les logiciels malveillants? :

Les logiciels malveillants sont capables d'infiltrer les réseaux et les appareils et sont conçus dans le but d'affecter négativement ces appareils, réseaux et/ou leurs utilisateurs [14].

Ces dommages peuvent se manifester de plusieurs manières pour l'utilisateur ou le terminal, selon le type de logiciel malveillant et son objectif. Les logiciels malveillants peuvent avoir des effets terribles dans certaines situations tout en ayant des effets très mineurs et inoffensifs dans d'autres [14].

Quelle que soit la technique, tous les logiciels malveillants sont créés pour exploiter les appareils aux dépens de l'utilisateur et en faveur du pirate, la personne qui a créé et/ou utilisé le logiciel [14].

1.5) Techniques d'obscurcissement :

Les attaquants utilisent des techniques d'obscurcissement, qui peuvent être classées en trois groupes et comprennent la compression (également appelée emballage), le polymorphisme et le métamorphisme, pour dissimuler leur code nuisible. Nous détaillerons ces différentes approches dans les sections qui suivent :

1.5.1) compression (packing) :

Pour sauvegarder le binaire, un packer peut simplement servir d'armure. Utiliser un packer plutôt que d'implémenter explicitement une protection dans le code est plus pratique pour les attaquants.

Cependant, les logiciels malveillants sophistiqués développés par des gangs de cybercriminels organisés utilisent des packers uniques ou intègrent des mesures de sécurité sophistiquées dans des fichiers dangereux.[15]

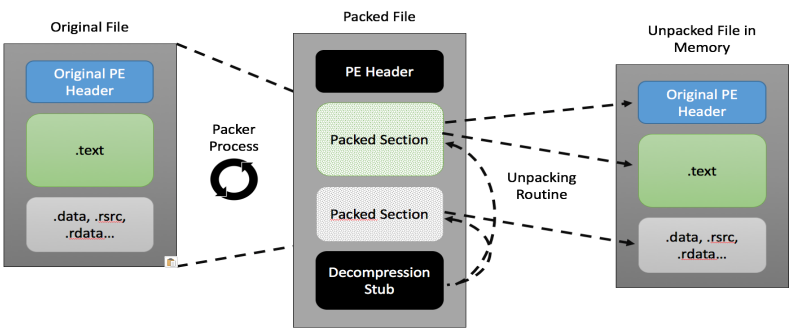


Figure 1.8 : Comment fonctionne packing [15]

Certains créateurs de logiciels malveillants utilisent des packers personnalisés, mais des packers commerciaux open source sont également utilisés. Certains emballeurs populaires incluent : **UPX - Themida - The Enigma Protector - VMProtect - Obsidium - MPRESS - Exe Packer 2.300 - ExeStealth …** [16]

1.5.2) Cryptage du code :

Pour rendre difficile leur identification par les moteurs antivirus, les virus polymorphes cryptent leur code de manière unique à chaque infection (ou à chaque génération d'infections). En raison de la capacité des logiciels malveillants polymorphes à se décrypter afin d'initier l'infection, la détection des logiciels malveillants basée sur l'émulation a été créée. Cela signifie que le chiffrement et le déchiffrement polymorphes doivent finalement aboutir à la production du vrai code en mémoire, là où l'émulation réussit. Les auteurs de logiciels malveillants ont créé des logiciels malveillants métamorphiques pour lutter contre cela, dans lesquels le code lui-même change à chaque infection [17].

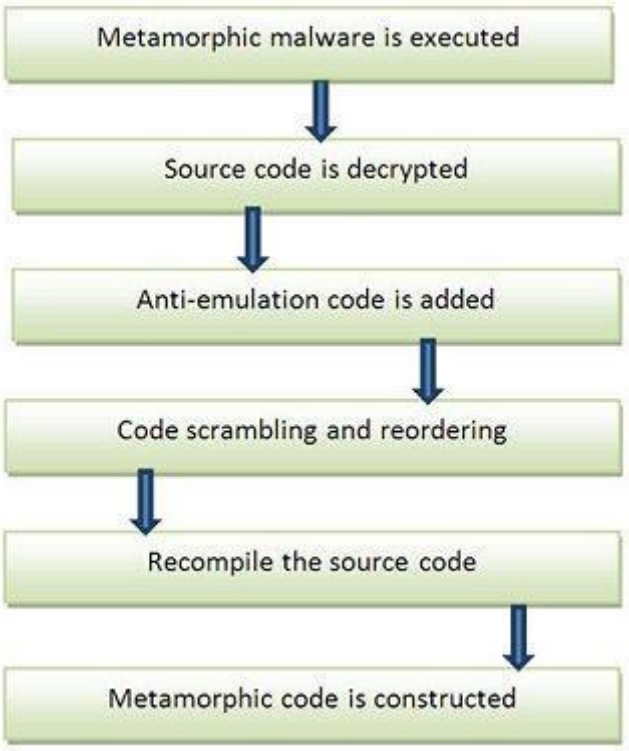


Figure 1.9 : Flux d'exécution des virus métamorphiques [18]

1.5.3) Mélanger les registres :

Une autre méthode par laquelle les auteurs de logiciels malveillants modifient la disposition des instructions dans le code est le brassage des registres. Le modèle de codage est modifié en mélangeant les registres, mais le code est toujours correctement exécuté. Cela sert principalement à obscurcir le vrai code, ce qui rend son interprétation difficile [18].

1.6) Conclusion :

Les logiciels malveillants en général, leurs variétés et leurs objectifs ont tous été couverts dans ce chapitre. Nous avons inclus des informations à leur sujet qui sont pertinentes pour notre enquête. Une technologie de détection qui nous a aidé à identifier différents types et familles de logiciels malveillants a pu identifier le volume incroyable de logiciels malveillants.

Dans le 2ème chapitre, nous présenterons les différentes techniques de détection des malwares, expliqueront leur fonctionnement et conclurons par une comparaison entre elles.

Chapitre 2
Systèmes de détection des logiciels malveillants

2.1) Introduction :

Un système de détection est nécessaire pour contrer des types particuliers de menaces informatiques. appareil de détection. Ce dernier offre les connaissances nécessaires pour développer des défenses et des techniques d'atténuation efficaces contre diverses infections. Dans ce deuxième chapitre, nous examinerons en détail les systèmes de détection des logiciels malveillants, notamment leur fonctionnement individuel et les différences entre leurs différentes philosophies de détection.

2.2) Définition :

Le système de détection des logiciels malveillants est basé sur des signatures, qui sont un ensemble de caractéristiques partagées par les fichiers malveillants, en l'occurrence les fichiers apk, qui sont utilisés pour installer des applications Android. La détection utilise principalement deux techniques - l'analyse statique et l'analyse dynamique - pour identifier les fichiers potentiellement dangereux. Chacune d'entre elles possède une méthode de détection unique [15].

Ces signatures de logiciels malveillants et profils comportementaux - ou, pour le dire autrement, l'ensemble des caractéristiques partagées par l'exécution d'échantillons de logiciels malveillants - sont créés par l'analyse, qu'elle soit statique ou dynamique. Les anomalies sont maintenant classées et trouvées à l'aide de techniques d'apprentissage [15].

Une application est analysée pour extraire des informations à son sujet, telles que les ressources qu'elle doit utiliser, son comportement et tout ce qui est prévu pour exécuter une certaine section de son code [15].

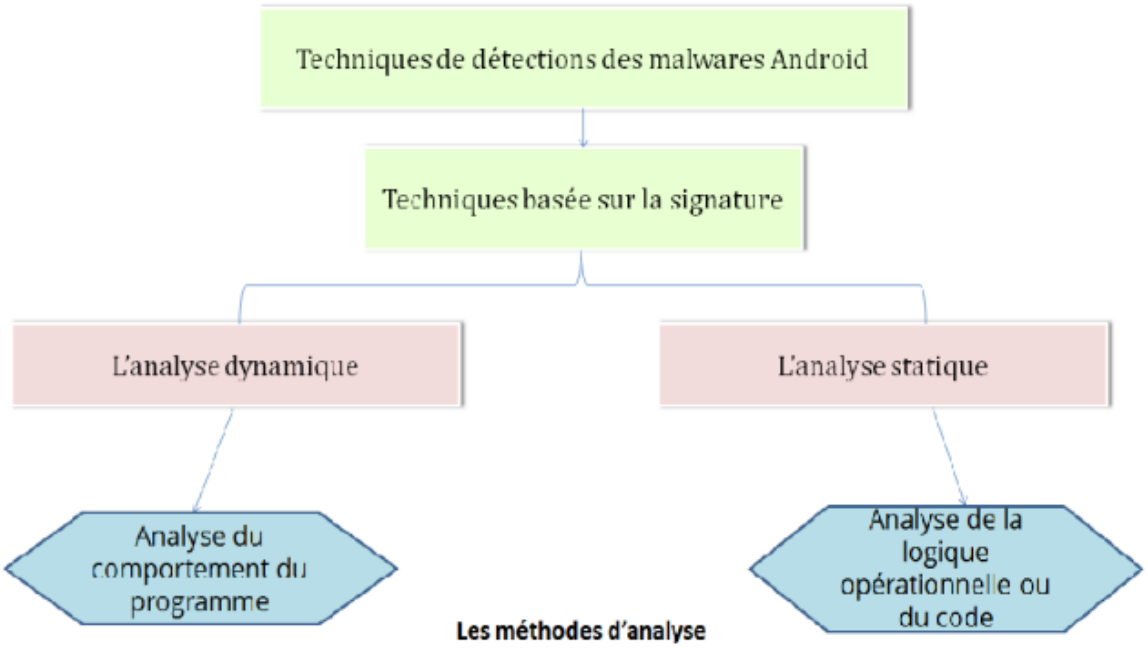


Figure 2.1 : Taxonomie du système de détection des malwares [16]

2.3) Analyse statique :

2.3.1) Definition :

L'analyse statique, souvent appelée analyse de code, est une méthode qui repose sur la décompilation d'un programme afin d'examiner son code source. En raison du temps d'analyse, qui peut être considéré comme une perte de temps pour évaluer la sécurité du programme, cette approche, également connue sous le nom de rétro-ingénierie, n'est pas entièrement efficace. Bien qu'il soit possible de découvrir des vulnérabilités côté client sans exécuter de code, cela est pratiquement impossible. Par conséquent, l'objectif de l'analyse du code est d'identifier les problèmes de sécurité plus ou moins visibles afin d'avoir une idée de la sécurité globale de l'application. Androguard ou ARE sont les deux meilleurs outils à utiliser pour mener ce type d'étude [17].

Le premier simplifie le processus d'analyse (permissions, instructions dangereuses, similarités entre 2 applications, etc.) et le second est un ordinateur virtuel qui possède une ancienne version du premier [17].

Par conséquent, il est impossible de déterminer si un programme peut ou non causer des problèmes à l'exécution en analysant les programmes, même en recherchant les fautes potentielles d'exécution [17].

Bien qu'il soit difficile de trouver toutes les fautes dans chaque programme, cela ne signifie pas que les techniques d'analyse statique doivent être entièrement abandonnées. La plupart du temps, il est facile d'identifier une partie non négligeable des problèmes présents [17].

Il faut utiliser des techniques qui fonctionnent admirablement bien sur la grande majorité des programmes réels. Par conséquent, il s'agit d'approximations. Ces procédures approximatives n'indiquent pas un manque de rigueur [40]. Les méthodes utilisées doivent être dignes de confiance, mais elles ne doivent pas nécessairement être les meilleures. Car il est concevable que le système présente encore certaines failles. Les méthodes d'analyse statistique ne seront pas en mesure de déterminer avec une certitude absolue si une erreur existe ou non. Par conséquent, il y a toujours une chance que les outils signalent des problèmes qui n'existent pas, des événements qui n'existent pas ou qu'ils ne trouvent pas les failles présentes dans le code. Les deux principales familles d'analyse statique sont l'interprétation abstraite et la vérification de modèle [17].

2.3.2) Outils d’analyse statique :

Voici quelques outils pour aider à l'analyse statique [18] :

- **PeStudio :** PeStudio est un outil fantastique à utiliser et fournit une tonne de détails, que vous appreniez à analyser les logiciels malveillants, à gérer un problème de logiciel malveillant ou à créer une règle YARA [19].
- **SonarQube:** (anciennement Sonar) SonarSource a créé la plate-forme open source pour effectuer des révisions automatiques avec analyse de code statique afin de trouver des défauts et des odeurs de code sur 29 langages de programmation [20].
- **PVS-Studio :** est un analyseur de code statique commercial qui prend en charge C, C++, C++11, C++/CLI, C++/CX, C# et Java. Il veille à la qualité, à la sécurité et à la sûreté du code (SAST) [21].
- **DeepSource :** L'approche la plus rapide et la plus fiable pour effectuer une analyse de code statique sur votre code est avec DeepSource [22].
- **Embold :** est un outil d'analyse de logiciels basé sur l'IA qui aide les équipes à analyser et à améliorer le calibre de leurs logiciels [23].

2.4) Analyse dynamique :

2.4.1) Definition :

L'exécution et le suivi des actions d'une application constituent l'analyse dynamique. L'analyse dynamique, par opposition à l'analyse statique, se concentre sur l'activité du programme, notamment les appels de fonction, les chaînes enregistrées en mémoire, le trafic créé, etc [17].

Lorsqu'il n'est pas possible d'accéder au code source du programme (légalité de la rétro-ingénierie), l'analyse dynamique est utilisée pour enquêter sur les logiciels malveillants (dans cette situation, l'utilisation d'un environnement sécurisé) [17].

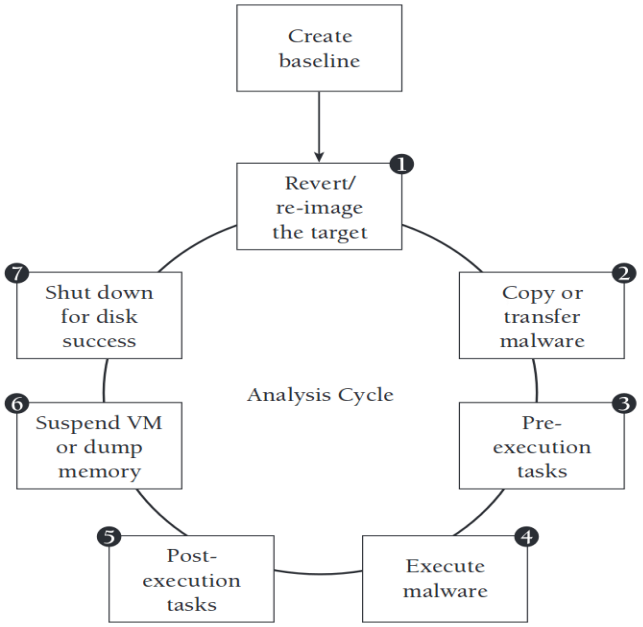


Figure 2.2 : Fonctionnement de la plate-forme d'analyse dynamique [24]

2.4.2) Outils d'analyse dynamique :

Voici quelques outils pour aider à l'analyse dynamique [25]:

- **CrowdStrike Falcon Insight EDITOR'S CHOICE :** Cet EDR suit le comportement des intrus en plus de mener une analyse à deux niveaux des

logiciels malveillants. Avec des modules de point de terminaison pour Windows, macOS et Linux, le système est basé sur le cloud [26].

- **Cuckoo Sandbox:** fournit un mélange entre des méthodes d'analyse de logiciels malveillants automatisées et humaines, ainsi que de nombreux paramètres de bac à sable [27].
- **IDA Pro :** un outil hautement technologique créé avec des experts en criminalistique et en cybersécurité à l'esprit [28].

2.5) Comparaison des méthodes d'analyse :

Le comportement avant et après l'exécution du programme, ainsi que les résultats de chaque approche, servent de base pour comparer les différentes techniques d'analyse. Ici, nous comparerons static et dynamic analyse à plusieurs égards, à savoir :

Une analyse : L'enquêteur n'a pas à passer par chaque étape du cycle puisque l'examen d'analyse statique est une méthode très simple et directe pour enquêter sur les tests de logiciels malveillants sans les exécuter.

D'autre part, un examen d'analyse dynamique implique une étude approfondie utilisant la conduite et les activités du test de malware pendant son exécution pour avoir une meilleure compréhension de l'exemple.

Technique impliquée : Une preuve identifiable intrigante pour le document parallèle est la double marque d'enregistrement du logiciel malveillant, qui est disséquée dans le cadre de l'analyse statique.

Mai L'analyse dynamique consiste à examiner le comportement des logiciels malveillants dans un environnement sandbox afin qu'il n'affecte pas les autres frameworks.

Approcher : L'analyse statique utilise une approche basée sur les marques pour traiter l'investigation et la découverte des logiciels malveillants.

L'analyse dynamique utilise une approche basée sur la conduite pour évaluer l'utilité du logiciel malveillant en tenant compte des actions entreprises par le virus spécifique.

Méthodologie : Des techniques d'analyse simples basées sur des remarques sont utilisées dans l'analyse statique.

Grâce à une analyse dynamique, les actions sont examinées plus en profondeur.

2.6) Conclusion :

Les outils d'analyse statique ou dynamique énumérés ci-dessus sont davantage basés sur des méthodes formelles. De nouveaux outils basés sur l'apprentissage sont en cours de développement. Ils sont utilisés pour catégoriser et trouver des logiciels malveillants. Ces techniques permettent aux ordinateurs d'apprendre à partir des données en utilisant des méthodologies mathématiques et statistiques.

Nous aborderons l'apprentissage automatique en cybersécurité dans le troisième chapitre et décrirons le fonctionnement de chacune de ces méthodes.

Chapitre 3

la détection des logiciels malveillants basée sur l'apprentissage profond

3.1) Introduction :

La représentation de notre environnement par les robots dans un avenir proche dépend en grande partie de notre capacité à utiliser avec succès l'intelligence artificielle (IA). Mais il est plus difficile qu'il n'y paraît de faire réfléchir les machines. L'apprentissage automatique (ML) est le seul moyen d'aider les robots à comprendre comme les humains et de développer une intelligence artificielle puissante qui peut nous aider dans divers domaines, y compris la cybersécurité.

3.2) l'apprentissage automatique pour la cybersécurité :

Pour relever les défis de la cybersécurité, des méthodes et techniques comprenant l'apprentissage automatique (machine learning), l'exploration de données, les statistiques et d'autres capacités interdisciplinaires ont été utilisées. Ces données sont disponibles via l'infrastructure réseau, les systèmes d'exploitation ou les systèmes d'information [29].

Le composant d'apprentissage profond du processus d'apprentissage automatique peut être utilisé pour identifier les logiciels malveillants en fonction de leur signature. Ces techniques de catégorisation et de prédiction peuvent être utilisées pour identifier des thèmes et des comportements communs à diverses cyberattaques, ce qui permet d'apporter des cyber réponses en temps réel. Elles sont capables de détecter les logiciels malveillants dès qu'ils se manifestent [29].

Les professionnels de la sécurité s'efforcent toujours d'obtenir des performances accrues de la part d'un système de détection qui présente le taux de détection le plus élevé et le taux de fausses alarmes le plus faible, car la collecte de données et le trafic réseau ont créé un défi en matière de big data. Les techniques d'apprentissage profond sont donc idéalement adaptées aux très grands ensembles de données. Afin de distinguer les comportements normaux des comportements aberrants et d'identifier les activités malveillantes ou potentiellement dangereuses, elles ont été créées pour la détection des anomalies du réseau [29].

3.3) L'apprentissage profond :

L'apprentissage profond fait partie d'un groupe d'approches d'apprentissage machine (ML). L'apprentissage profond donne de bons résultats dans de nombreuses tâches d'intelligence artificielle (IA) par rapport aux techniques traditionnelles d'apprentissage

machine (ML). Les modèles profonds utilisent une variété de conceptions relativement nouvelles qui font appel à plusieurs étapes non linéaires de traitement de l'information. Les informations sont traitées dans des couches hiérarchiques, chaque couche recevant et interprétant les données de la couche précédente dans le but d'apprendre des représentations de données [30].

L'architecture des réseaux profonds est généralement organisée en couches de neurones : une couche d'entrée, une ou plusieurs couches cachées.
la couche de sortie.
Chaque paire de niveaux adjacents est liée. Les poids sont les relations qui existent entre eux. Il n'existe aucune relation entre les "neurones" (également appelés "nœuds") d'une même couche. L'architecture typique d'un modèle de réseau neuronal profond est illustrée à la figure 3.1.

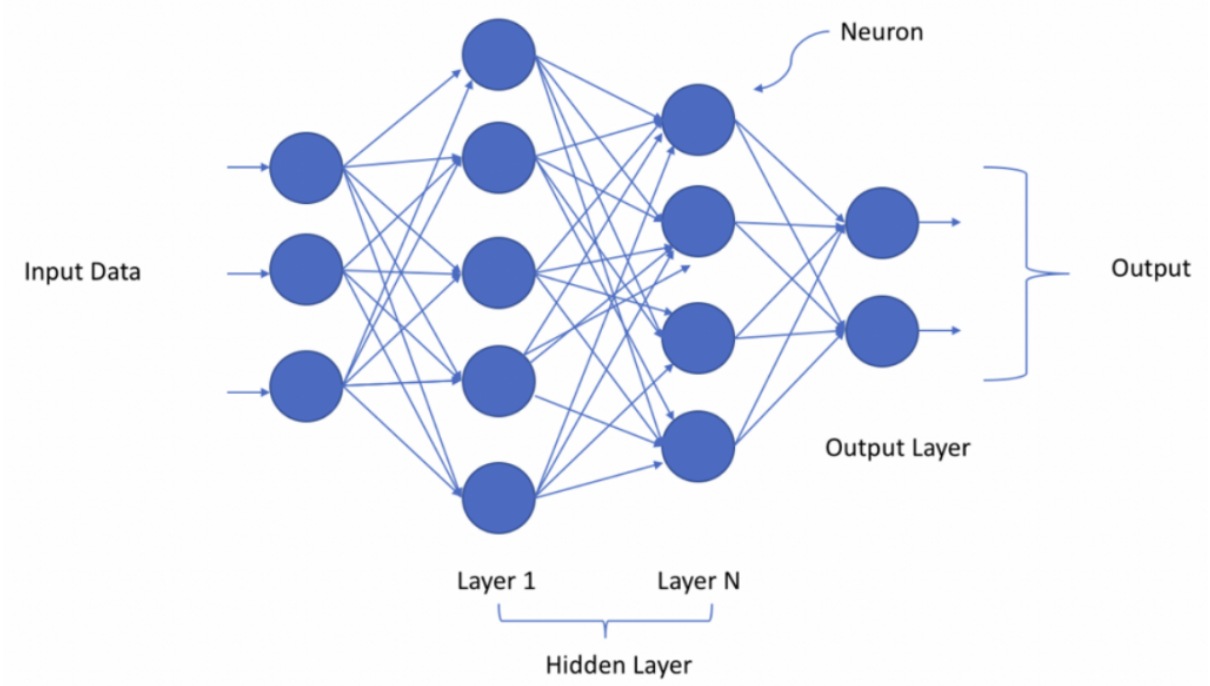


Figure 3.1 : L'architecture d'un modèle deep learning [31]

3.3.1) les réseaux de neurone convolutif (CNN) :

Le réseau neuronal convolutif (CNN) est surtout utilisé dans le domaine du traitement de l'image . La figure 3.2, qui omet tous les liens, les niveaux cachés et ses unités, l'illustre. Les CNN excellent dans les tâches de vision par ordinateur et dépassent tous les autres algorithmes traditionnels de ML. Ils ont plusieurs utilisations dans le traitement des images et des vidéos, le traitement du langage naturel (NLP), les systèmes de recommandation, etc [32].

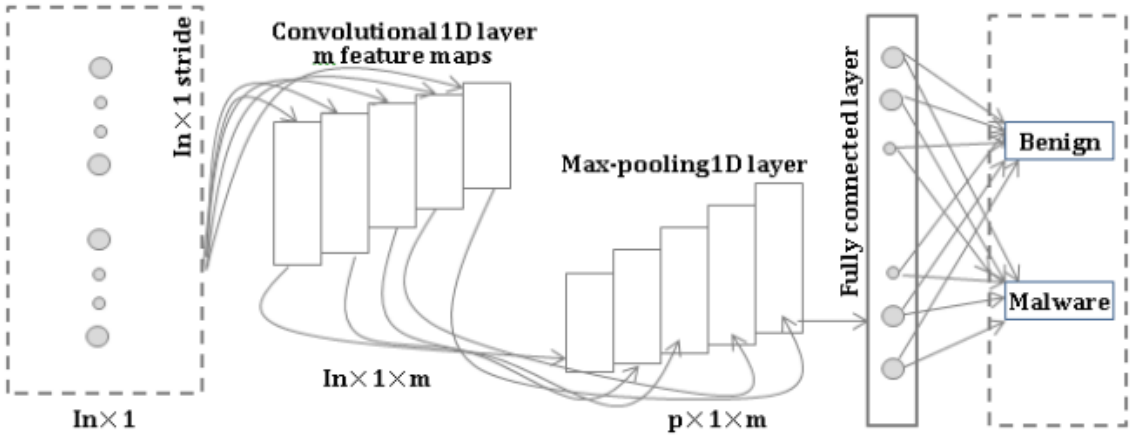


Figure 3.2 : Architecture du CNN pour la détection des logiciels malveillants [32]

M : le nombre de filtres.
Ln : le nombre de caractéristiques d'entrée.
P : la dimension réduite des dimensions réduites des caractéristiques.

Les couches convolutionnelles, des couches entièrement connectées et des couches groupement (Pooling) sont trois types différents de couches spéciales qui rendent les réseaux convolutionnels exceptionnellement efficaces [33].

Convolution layer : Les caractéristiques de haut niveau sont extraites par convolution. Elle est constituée d'un certain nombre de filtres d'apprentissage (également appelés noyaux), dont chacun utilise le volume d'entrée pour représenter un attribut indépendant différent. Le processus de feed forward convolue chaque filtre sur la largeur et la hauteur du volume d'entrée, en calculant le produit des points entre les entrées et les valeurs du filtre pour produire une nouvelle carte de caractéristiques qui représente l'information. Ces filtres sont constitués d'une couche de poids de connexion, ont un petit champ de réception (la taille du noyau) et sont constitués d'une couche de poids de connexion. Par conséquent,

le réseau acquiert des connaissances sur les filtres qui s'activent lorsqu'il identifie un type d'élément crucial et particulier à un certain emplacement géographique dans l'entrée [33].

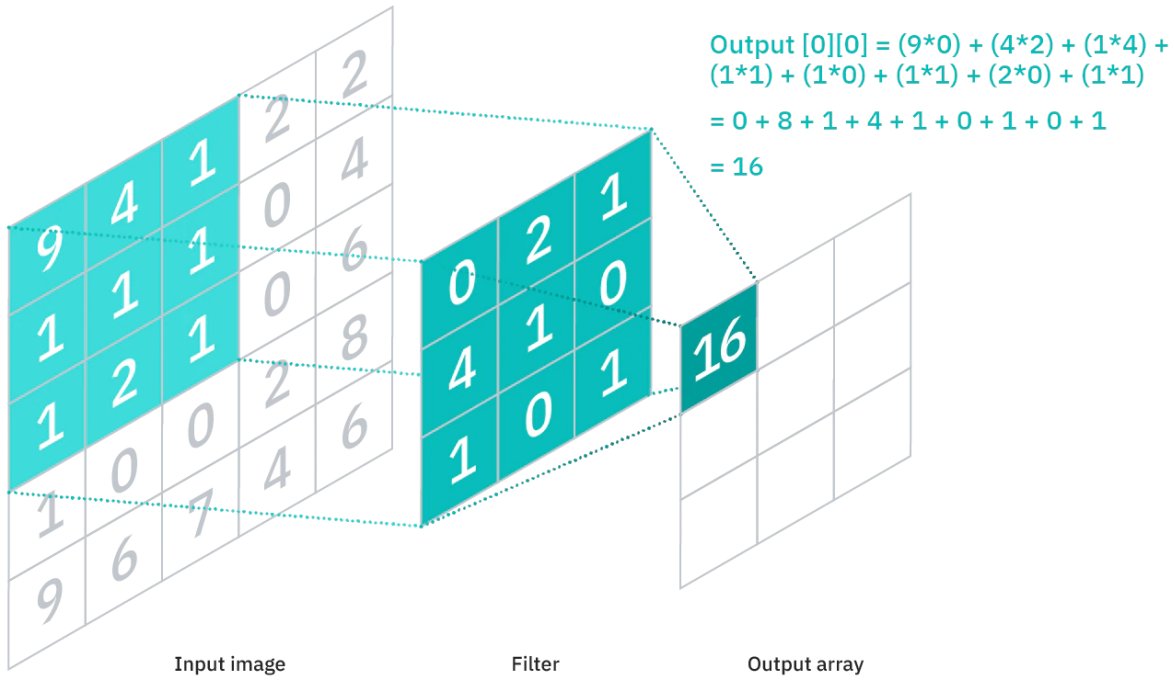


Figure 3.3 : Convolution layers [34]

Le nombre de paramètres requis pour le processus de convolution est considérablement réduit car chaque couche convolutive utilise le même noyau de convolution. Après chaque couche convolutive, une fonction d'activation non linéaire sera utilisée. Fonction d'activation ReLU (Rectified Linear Units) dans les CNN profonds [33].

Pooling layer : La procédure de pooling agrège l'activité des neurones d'une couche en un seul neurone de la couche suivante après la transformation ReLU. La couche de mise en commun fonctionne indépendamment sur chaque élément d'entrée et permet de réduire progressivement la taille de la représentation afin de diminuer le nombre de paramètres ou de poids. Cela réduit le coût de calcul du réseau tout en protégeant les données les plus importantes. En outre, il permet de contrôler l'apprentissage. Il peut utiliser l'une des deux techniques de mise en commun [33] :

-La mise en commun maximale (Max-Pooling) : Utilise la valeur maximum de chaque ensemble de neurones de la couche précédente [33].

-La mise en commun moyenne (Average-Pooling) : Utilise la valeur moyenne de chaque ensemble de neurones de la couche précédente dans le pooling moyen Comme la convolution [33].

Le pooling est une technique de sous-échantillonnage non linéaire. Selon la figure 3.4, l'opération de Max-Pooling avec une entrée 1D et un noyau de taille 2 produit une seule valeur de sortie qui est la valeur maximale pour le Max-Pooling ou la valeur moyenne pour le Average-Pooling. Le noyau de mise en commun effectue une convolution sur le volume d'entrée et le divise en un ensemble de régions non chevauchantes [33].

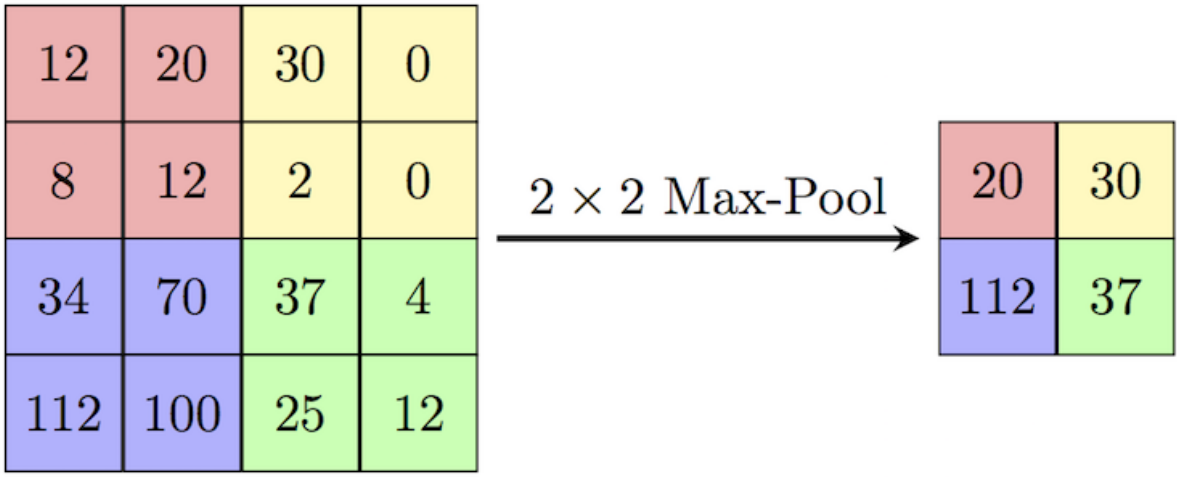


Figure 3.4 : Pooling layers [35]

Fully connected layers : Il y a une ou plusieurs couches complètement liées à la fin d'un CNN (chaque nœud de la première couche est connecté à chaque nœud de la couche suivante). Elles comprennent la réalisation d'une classification à l'aide d'extractions de caractéristiques convolutionnelles. Une fonction d'activation Softmax, présente dans la dernière couche, crée une valeur de probabilité entre 0 et 1 pour chaque étiquette de classe que le modèle tente de prédire. Les couches multiples de mise en commun des moyennes peuvent remplacer les couches entièrement liées dans certaines topologies modernes de CNN. Cela permet de réduire considérablement le nombre total de paramètres et d'améliorer la prévention du sur-apprentissage dans ces réseaux [33].

3.4) Les Data-sets d'évaluation de détection des malwares basé sur Deep learning :

Les Data-sets utilisés dans les travaux publiés pour l'application de l'apprentissage approfondi dans la cybersécurité jouent un rôle essentiel pour la validation de toutes approches DL proposées.

Dataset	Malware Time	Family	# Families	# Samples	# Benign	# Malware	Malware Binaries	Feature Vectors
Microsoft	N/A (Before 2015)	●	9	10,868	0	10,868	◑	○
Ember	01/2017–12/2018	◑	N/A	2,050,000	750,000	800,000	○	●
UCSB-Packed	01/2017*–03/2018	○	N/A	341,445	109,030	232,415	●	○
SOREL-20M	01/2017–04/2019	○	N/A	19,724,997	9,762,177	9,962,820	●	●
BODMAS	**08/2019–09/2020**	●	**581**	134,435	77,142	57,293	●	●

Tableau 3.1 : Ensembles de données public relatives à la détection des malwares [36]

3.5) Comparaison entre deep learning et machine learning :

Entre l'apprentissage automatique et l'apprentissage profond, il existe des distinctions importantes. Les cinq différences les plus importantes, selon [W8], sont les suivantes :

1. L'implication humaine : Pour que l'apprentissage automatique produise des résultats, une implication humaine plus fréquente est nécessaire. implication constante des personnes pour obtenir des résultats. Bien que plus difficile à mettre en place, l'apprentissage profond nécessite moins d'intervention une fois qu'il est en marche. une post-intervention minimale est nécessaire [37].

2. Le matériel : Alors que les systèmes d'apprentissage profond ont besoin de matériel et de ressources nettement plus puissants, les algorithmes d'apprentissage automatique sont fréquemment moins sophistiqués et peuvent fonctionner sur des PC standard. Le besoin de plus de puissance a poussé l'adoption des GPU [37].

3. Temps : les systèmes d'apprentissage automatique peuvent être rapidement mis en place et exécutés, mais leur efficacité peut être limitée [37].

4. Méthode : L'apprentissage automatique utilise souvent des techniques classiques traditionnelles comme la régression linéaire et tend à exiger des données organisées. L'apprentissage profond utilise des réseaux neuronaux et est destiné à gérer d'énormes quantités de données non structurées [37].

5. Applications : Votre banque, votre cabinet médical et votre compte de messagerie électronique utilisent tous l'apprentissage automatique. La technologie d'apprentissage profond permet la mise en place de programmes autonomes de plus en plus sophistiqués, tels que les véhicules à conduite autonome ou les robots chirurgicaux [37].

3.6) Travaux connexes pour la détection des malwares basé sur le DL :

L'apprentissage en profondeur, contrairement aux méthodes d'apprentissage automatique, est récemment entré dans le domaine de l'identification des logiciels malveillants.

l'une des premières recherches est celle de Tobiyama [38] ., il combine les réseaux récurrents avec les réseaux convolutifs . Ils ont utilisé un analyseur de séquences pour examiner les séquences d'appels d'API basées sur des réseaux de neurones récurrents. La sortie du RNN est transformée en une image que CNN a traitée. Deux couches de convolutions constituaient la conception CNN, et une opération de mise en commun venait après chaque couche .

et en 2018, INVIDIA a sorti des pièces très créatives. Raff, Zak and all [39] ont suggéré une technique pour repérer les logiciels malveillants en examinant le flux d'octets de l'exécutable. Cette méthode est basée sur l'analyse de séquences et utilise directement les données brutes (Raw Bytes). Ils ont pu développer un analyseur de séquences d'une capacité de 2 millions de pas de temps, qui a fait progresser l'état de l'art dans le domaine du traitement des séquences et de la détection des logiciels malveillants .

La même année, Zhou a proposé un modèle basé sur les RNN et les CNN qui combine des propriétés statiques et dynamiques [40]. Les histogrammes d'octets, les histogrammes d'entropie, les informations de section, les informations d'importation et d'exportation, les en-têtes et les chaînes sont des exemples de données statiques. Le RNN a utilisé les appels d'API comme caractéristique dynamique pour l'analyse. Les informations sont utilisées pour créer une image exécutable, qu'un CNN examinera ensuite .

3.7) Conclusion :

Pour détecter les logiciels malveillants, diverses méthodologies et architectures d'apprentissage en profondeur ont été utilisées. Les ensembles de données et les caractéristiques d'entrée sélectionnés affectent les performances de ces algorithmes d'apprentissage en profondeur proposés. Cependant, l'utilisation des mêmes stratégies et processus d'apprentissage pour une gamme de différentes classes de menaces potentielles ne garantit pas nécessairement les mêmes résultats et c'est là que l'apprentissage en profondeur émerge.

Alors que l'apprentissage en profondeur a une application très large. Il se développe rapidement, avec de nouveaux modèles, algorithmes et variations apparaissant chaque semaine. Pour les chercheurs en sécurité, l'utilisation de nouvelles technologies et les tests de performance de diverses architectures d'apprentissage en profondeur existantes restent des domaines de recherche clés.

Chapitre 4
Conception de la Solution proposé

4.1) Introduction :

Alors que les types de logiciels malveillants continuent de croître, les scanners antivirus Incapables de répondre aux exigences de protection, des millions d'hôtes sont attaqués. Alor Protéger votre système informatique des logiciels malveillants est donc une mesure de sécurité.

Les tâches de cybersécurité les plus importantes pour les utilisateurs individuels et les entreprises Même une seule attaque peut entraîner une compromission des données et une perte substantielle. À Pour cette raison, des techniques basées sur l'apprentissage automatique peuvent être utilisées.

l'objectif de ce projet est de développer la preuve de concept pour la classification des logiciels malveillants basée sur l'apprentissage automatique avec de puissants classificateurs. meilleure caractéristique Déterminé et extrait de toutes les propriétés données. Algorithme le plus précis pouvez identifier les fichiers malveillants avec les taux d'erreur les plus faibles. approcher les performances La méthode de validation proposée a été évaluée en tenant compte de diverses mesures d'évaluation algorithmes d'apprentissage en profondeur . précision, rappel, score F1, Taux de détection et taux de fausses alarmes.

4.2) Environnement de développement :

L'exigence d'avoir certaines ressources matérielles dans le Deep learning est obligatoire pour faire des calculs intenses. Cela nous a conduit vers le Cloud .

Le Cloud fournit pour nous des ressources de calculs et de mémoires importantes qui dépassent nos ordinateurs.

Google Collaboratory : Colab est un environnement de notebook Jupyter gratuit qui fonctionne entièrement dans le nuage. Il ne nécessite pas de configuration et les notebooks que vous créez peuvent être édités simultanément par les membres de votre équipe - exactement comme vous éditez des documents dans Google Docs [41]. Google Colab prend en charge de nombreuses bibliothèques d'apprentissage automatique populaires qui peuvent être facilement chargées dans votre notebook. Il nous offre un processeur GPU gratuit, (13 Go de RAM et plus de 107 Go de stockage).

Python : Python est un langage de programmation interprété, orienté objet, de haut niveau et à sémantique dynamique. La syntaxe de Python est simple et facile à apprendre, privilégie la lisibilité et réduit donc le coût de la maintenance des programmes. Python prend en charge les modules et les packages, ce qui encourage la modularité des programmes et la réutilisation du code [49].

TensorFlow : une bibliothèque open source créée par Google, permettant de développer et d'exécuter des applications de Machine Learning pour effectuer des opérations numériques complexes sur plusieurs plateformes comme Les GPU et Les CPU [50].

Keras : Keras offre des fonctionnalités minimales mais très productives grâce à sa bibliothèque d'apprentissage profond. Keras est écrit en Python comme une API d'apprentissage profond et il fonctionne au-dessus de TensorFlow, une plateforme d'apprentissage automatique. Keras a été développé pour permettre une expérimentation rapide. Il est facile à aborder et offre une interface très productive pour résoudre plusieurs problèmes d'apprentissage automatique en se concentrant sur une approche moderne de l'apprentissage profond. L'avantage essentiel de Kera est qu'il peut prendre l'idée d'un développeur et le guider vers des résultats définitifs [51].

Spyder : Spyder est un environnement scientifique puissant écrit en Python, pour Python, il est conçu par et pour des scientifiques, ingénieurs et analystes de données. Il offre une combinaison unique des fonctionnalités avancées d'édition, d'analyse, de débogage et de profilage d'un outil de développement complet avec l'exploration de données, l'exécution interactive. On a utilisé Spyder dans le pré traitement des donnés [52].

4.3) Dataset :

La data-set **Malware Detection PE-Based Analysis Using Deep Learning Algorithm Dataset** est l'ensemble de données choisi pour cette étude. Publiée par Anh et al [42]. Elle est constituée de plusieurs types de malwares. Ces types sont des Fichier exécutable (.exe)
Benign 1000
Loker 330
Mediyes 1450
Winwebsec 4040
Zbot 2100
Zeroaccess 690

Tous les logiciels malveillants sont collectés par virusshare [43] et malicia-project [44]. Les fichiers bénignes ,sont extraits de logiciels légitimes installés de différentes catégories de windows ,tous les logiciel malveillant sont vérifiés par VirusTotal [43] pour s'assurer que chaque fichier appartient à son type.

4.4)Préparation des données :

Dans notre travail nous avons besoin de deux grandes étapes. La première étape consiste à extraire le code binaire à partir des fichiers exécutable avant de les transformer en image au niveau gris. La deuxième étape permet de résoudre le problème du déséquilibre des classes de données et les différentes étiquettes.

Les Classes concernés	Type	Nb d'instances pour l'apprentissage	Nb d'instances pour le test
Benign	Benign	774	305
Loker	Ransomware	264	83
Mediyes	Trojan	1160	438
Winwebsec	Adware	2464	908
Zbot	spyware	1176	341

Zeroaccess	botnet	552	216

Tableau 4.1 : résultat de nettoyage

4.4.1) Nettoyage de données :

Dans cette partie , nous allons explorer les octets bruts d'un fichier EXE afin de le convertir en image. Le code binaire de tous les fichiers de même classe seront lit et convertit chaque octet en base 256 avant les placés dans un fichier csv pour supprimer les redondances

```
f = open('C:\\Benign\\png\\inpute', 'w', newline='')
writer = csv.writer(f)
for folderName, subfolders, filenames in os.walk('C:\\Benign\\'):
    for filename in filenames:
        if filename.endswith('.exe'):
            binary = getBinaryData('C:\\Benign\\' + filename)
            print("Shape of lista is : "+str(np.shape(binary)))
            writer.writerow(binary)

f.close()
with open('C:\\Benign\\png\\inpute', 'r') as in_file, open('C:\\Benign\\png\\final', 'w') as out_file:
    seen = set()
    for line in in_file:
        if line in seen: continue

        seen.add(line)
        out_file.write(line)
```

Figure 4.1 : Example sur la classe Benign

4.4.2) La transformation des données :

Le flux d'octets d'un exécutable sera transformé en image. Par exemple, la séquence d'octets : E4 C0 56 A3 D2 78 56 A3 FF, peut être représentée par la matrice suivante :

E4	C0	56
A3	D2	78
56	A3	FF

Figure 4.2 : Image en niveau de gris pour une séquence d'octets

Lire le fichier csv ligne par ligne et on va créer une image à l'aide de la bibliothèque PIL. Au début on calcule la taille de l'image a créé :

```
41
42  def get_size(data_length, width=None):
43
44      if width is None:
45
46          size = data_length
47
48          if (size < 10240):
49              width = 32
50          elif (10240 <= size <= 10240 * 3):
51              width = 64
52          elif (10240 * 3 <= size <= 10240 * 6):
53              width = 128
54          elif (10240 * 6 <= size <= 10240 * 10):
55              width = 256
56          elif (10240 * 10 <= size <= 10240 * 20):
57              width = 384
58          elif (10240 * 20 <= size <= 10240 * 50):
59              width = 512
60          elif (10240 * 50 <= size <= 10240 * 100):
61              width = 768
62          else:
63              width = 1024
64
65          height = int(size / width) + 1
66
67      else:
68          width  = int(math.sqrt(data_length)) + 1
69          height = width
70
71      return (width, height)
72
```

Figure 4.3 : La fonction getsize pour avoir la taille d'image

Puis on sauvegarde le flux d'octet dans l'image

```
99  name = 1
100 file = open('C:\\Benign\\png\\final')
101 csvreader = csv.reader(file)
102 for row in csvreader:
103     name = 'C:\\Benign\\' + str(name)
104     greyscale_data = next(csvreader)
105     size = get_size(len(greyscale_data), None)
106     image = Image.new('L',size)
107     image.putdata(greyscale_data)
108     dirname     = os.path.dirname(name)
109     name, _     = os.path.splitext(name)
110     name        = os.path.basename(name)
111     imagename   = dirname + os.sep + 'png' + os.sep + name + '_'+ '.png'
112     os.makedirs(os.path.dirname(imagename), exist_ok=True)
113     img_path = imagename
114     image.save(imagename)
115     print('The file', imagename, 'saved.')
116     name += 1
117
```

Figure 4.4 : Création d'une image

4.4.3) balancer les données :

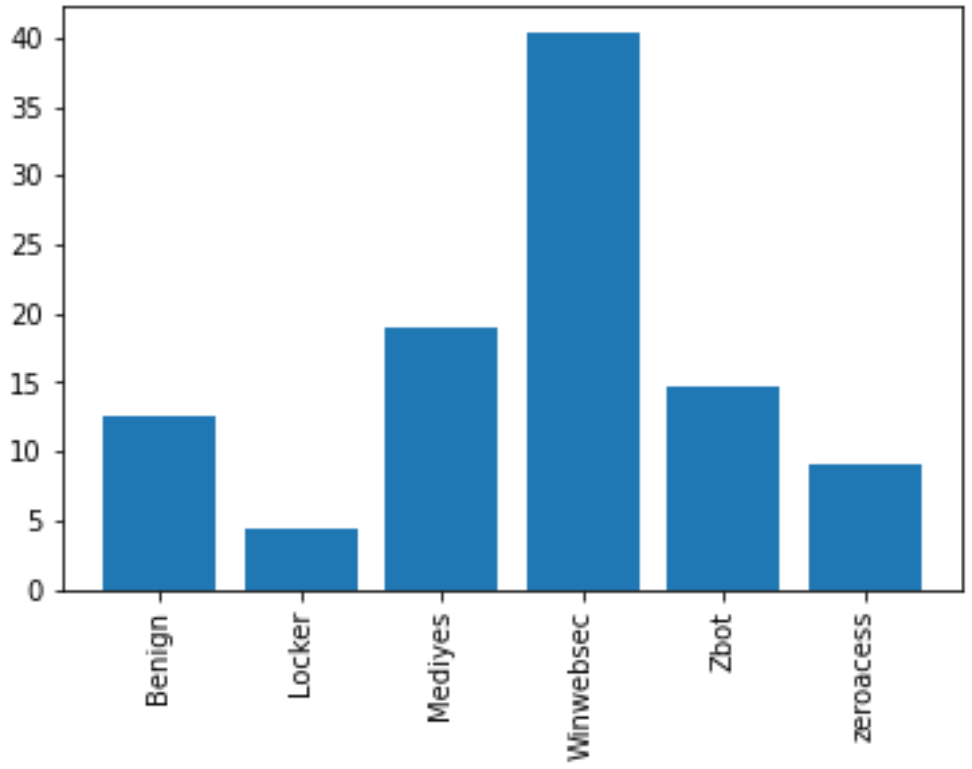


Figure 4.5 : Le pourcentage de données

Comme nous l'observons ici, l'ensemble de données est assez déséquilibré, la majorité des classes ont moins de pourcentage de classe Winwebsec , ce qui nous conduira à plus de faux positifs de cette classe avec un score f1 égal à zéro.

Nous essayons donc davantage de supprimer le déséquilibre des classes.
Dans notre cas, nous ne pouvions pas utiliser l'augmentation des données, en rééchantillonnant certaines classes car les images de notre ensemble de données sont des images de logiciels malveillants, de sorte que ces techniques ne nous aident pas.

Nous avons donc utilisé la technique des pondérations des classe,qui ce trouve dans la bibliothèque sklearn.utilis.class_weights

La fonction utilise les valeurs de y pour ajuster automatiquement les pondérations inversement proportionnelles aux fréquences de classe dans les données d'entrée. Pour utiliser cette méthode, y_train ne doit pas être codé a one-hot encoded.

Cette méthode donne moins de poids aux classes apparaissant fréquemment et plus de poids aux classes moins présentes lors du calcul de la propagation vers l'avant et vers l'arrière.

```
[24] from sklearn.utils import class_weight
     #Deal with unbalanced Data
     class_weights = class_weight.compute_class_weight(class_weight = 'balanced',
                                                       classes = np.unique(y_train_new),
                                                       y = y_train_new)

[25] class_weights.dtype

     dtype('float64')

[26] keys = [x for x in range(6)]

     mydict = {k:v for(k,v) in zip(keys,class_weights)}
     mydict

     {0: 1.2939042089985486,
      1: 3.8098290598290596,
      2: 0.8783251231527094,
      3: 0.41484411354118195,
      4: 1.1532988357050453,
      5: 1.8231083844580778}
```

Figure 4.6 : Pondérations des classe

Cette étude est basée sur les caractéristiques de code binaire extraites a partire des fichier exécutable. Nous avons évalué le taux de détection et le taux de fausse alarme ainsi que d'autre métrique de classification pour l'approche proposée .

4.5) L’architecture du modèle :

Avec la variété qui existe dans les techniques de deep learning, On a implémenté notre modèle avec Le CNN 1D (Le réseaux de neurone convolutif d’une seule dimension). On a utilisé cette technique car nos données sont des séries numériques avec une forme d’une seule dimension (forme textuel numérique). Notre Modèle est implémenté comme suite :

-Nous avons choisi la fonction ReLU Comme une fonction d’activation. Comme il y a aussi plusieurs fonctions (Tanh, sigmoid …etc.) mais les meilleurs résultats sont toujours obtenus avec «ReLU»

-Nous avons constaté qu’on a un problème de sur-apprentissage (Notre modèle est entraîné beaucoup avec les données d'entraînement. Donc le ‘Accuracy’ des données de test est réduit) ce problème est nommé OVERFITTING. C’est pour ça que nous avons utilisé la technique ‘Dropout’ pour combiner le pourcentage de l’Accuracy des données d'entraînement et les données de la validation.

-Pour la fonction de perte (Loss) nous avons utilisé la fonction ‘Binary-cross-entropy’.

-Nous avons aussi utilisé l’optimiseur ‘Adam’ avec un taux d’apprentissage 0.001 (Learning Rate = 0.001) au lieu de l’algorithme stochastique gradient descendant SGD (dans notre cas le SGD est donné un mauvais résultat). Le rôle de ‘Adam’ est de mettre à jour les poids d’un réseau de neurones afin de réduire la perte avec le minimum d'erreurs. Voici un exemple de notre modèle dans Google Collab :

```
[100] def malwaremodel():
          Malware_model = Sequential()
          Malware_model.add(Conv1D(12, (5), padding='same',input_shape=(64,64,1), activation = 'relu'))
          Malware_model.add(Conv1D(12, (5), activation = 'relu'))
          Malware_model.add(MaxPooling2D(pool_size=(2, 2)))
          Malware_model.add(Conv1D(12, (3), padding='same', activation = 'relu'))
          Malware_model.add(Dropout(0.25))
          Malware_model.add(Conv1D(12, (3), activation = 'relu'))
          Malware_model.add(Dropout(0.25))
          Malware_model.add(MaxPooling2D(pool_size=(2, 2)))
          Malware_model.add(Dense(128, activation='relu'))
          Malware_model.add(Flatten())
          Malware_model.add(Dense(1024, activation = 'relu'))
          Malware_model.add(Dense(num_classes, activation='softmax'))
          Malware_model.compile(loss='categorical_crossentropy', optimizer=Adam(learning_rate=0.001), metrics=['accuracy'])
          return Malware_model
```

Figure 4.7 : L'implémentation du model

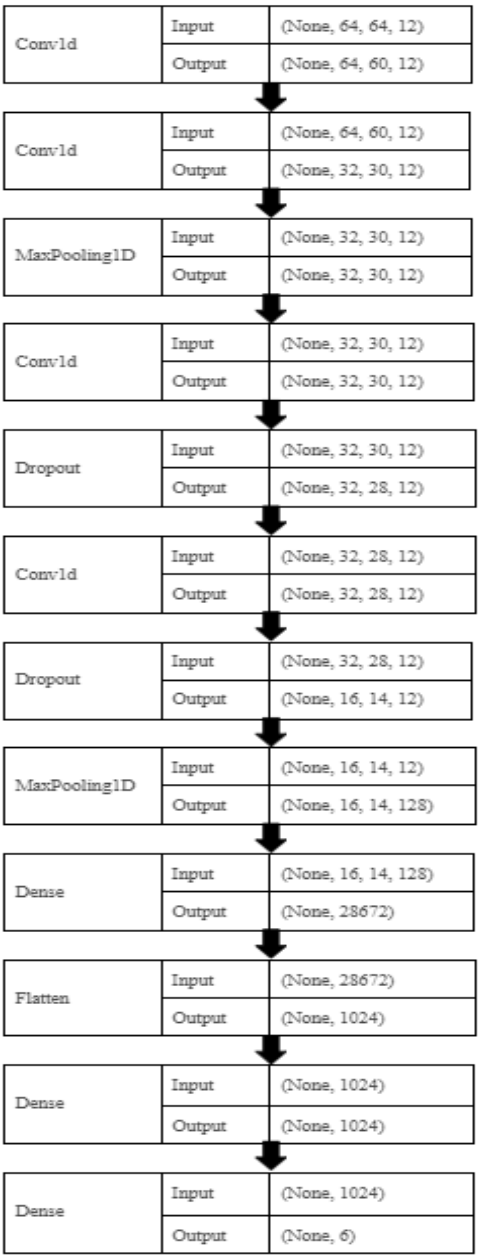
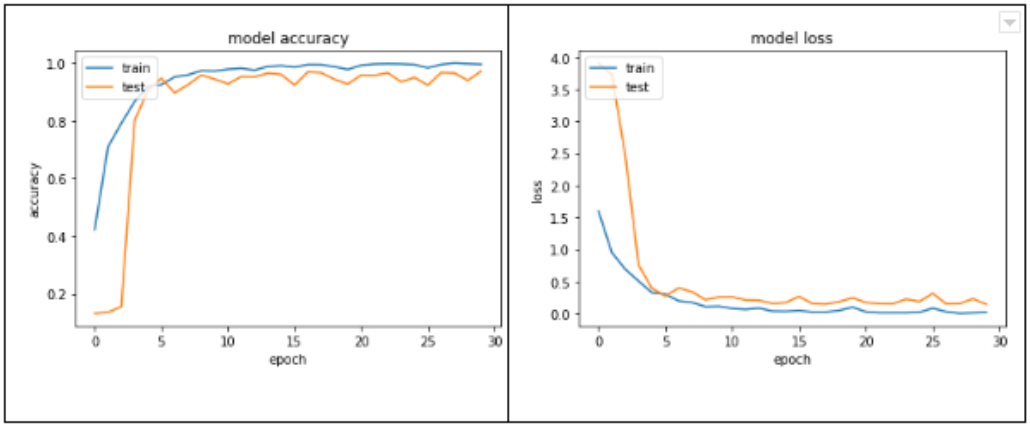


Figure 4.8 : L'architecture de model

4.6) Implémentation et Résultats :

Cette étude est créée dans le but de développer un framework efficace pour la détection et la classification des logiciels malveillants. Notre but est de maximiser les exactitudes possibles et aussi de minimiser le taux d'erreurs et des fausses alarmes. Tout d'abord, plusieurs tests ont été faits afin d'obtenir les bons paramètres pour ce modèle. Ces paramètres (Hyper-paramètres) ne peuvent pas être ajustés durant la phase de l'apprentissage et pourtant, ils ont un grand impact sur les performances de modèle durant l'apprentissage. Ils comprennent les variables qui déterminent la structure du réseau (Nbr de neurones, Nbr de couches, fonction d'activation, . . .), le lot d'échantillons (Batch Size) et le nombre d'itérations ...etc. Lorsqu'on arrive à un bon modèle avec le minimum de taux d'erreur et le maximum d'exactitude, nous avons ensuite testé ce modèle sur le sous-ensemble de test.

Les résultats sont présentés dans les figures 4.9 On a obtenu un bon résultat comme une première expérience (Accuracy = 99,58%, Loss = 0.14%). Ces résultats montrent que l'apprentissage profond convient à la caractérisation des logiciels malveillants. Le modèle a été évalué directement sur l'ensemble de tests. Ce dernier a été formé sur 27 itérations et nous notons ici que le modèle converge vers une valeur de perte minimale. Il a presque la même valeur de perte lors de l'apprentissage et l'évaluation. Ce qui indique que ce modèle sera généralisé bien au-delà de l'ensemble d'apprentissage. Ensuite, nous avons testé ce modèle sur l'ensemble de tests.




```
[296   5   0   0   2   2]
[ 22  48   3   4   2   4]
[  4   2 430   0   0   2]
[  0   2   1 904   0   1]
[  4   1   0   0 335   3]
[  1   0   0   1   0 214]]
```

Matrice de Confusion

Figure 4.9 : L'évaluation du modèle CNN

Pour avoir une autre perspective sur notre solution, on a essayé la solution proposée sur deux autre modèles INCEPTIONV3 [45] et VGG16 [46] de la bibliothèque keras pré-traine sur la dataset image-net [47] avec la technique L'apprentissage par transfert (transfert learning) [48] on change que le dernière couche pour avoir 6 classe a la sorte de modèle comme suite :

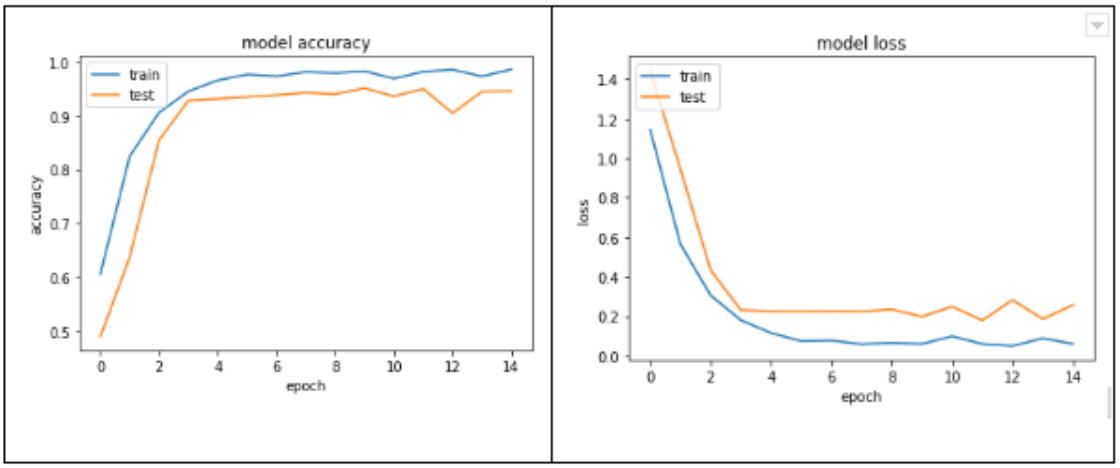
```
[ ]  x=Flatten()(vgg.output)

[ ]  pred=Dense(6,activation='softmax')(x)

[ ]  from keras.models import Model

[ ]  model=Model(inputs=vgg.input,outputs=pred)
```

Figure 4.10 : Changement de la dernière couche du model

pour les hyper-parameter on a utilisé ‘categorical_crossentropy’ comme fonction de loss,pour l'optimiser c’est Adam avec un learning rate = 0,0001, et 11 itération pour VGG16 et 15 pour INCEPTIONV3 les résultat sur l'ensemble de teste sont :

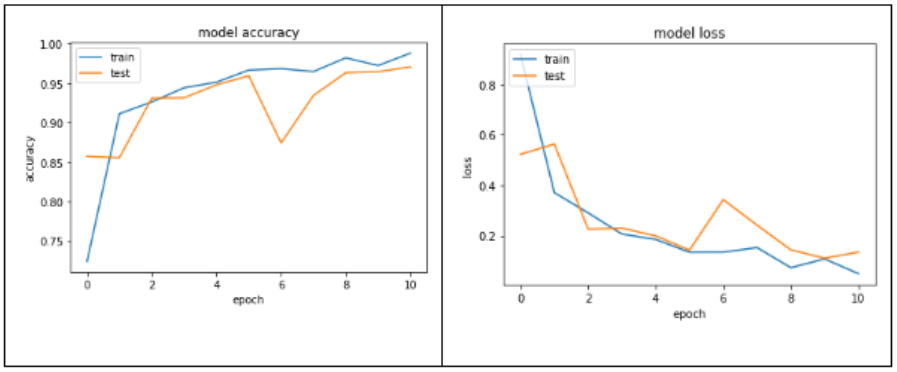
	Accuracy	Validation
INCEPTIONV3	94.64%	0.25%
VGG16	97.08%	0.13.%
CNN1D	97.12%	0.21.%

Tableau 4.2 :Les résultat sur les données de test




```
[275  12   2   1   1   5]
[ 13  83   1  10   0   6]
[  3   0 410   4   6   8]
[  0   2   4 886   0  14]
[  2   5   4   1 323   2]
[  1   2  10   2   2 193]]
Matrice de Confusion
```

Figure 4.11 : L’évaluation du modèle INCEPTIONV3




```
[258  10   2   0   0   0]
[ 14  68   4   4   0   4]
[  4   3 425   1   0   1]
[  0   1   2 922   0   1]
[  4   2   1   2 349   3]
[  1   3   0   0   0 204]]
```

Matrice de Confusion

Figure 4.12 : L'évaluation du modèle VGG16

5.7) Les mesures d'évaluation d'un modèle :

Précision (Pr) : le pourcentage des classe identifiées comme la meme classe TP parmi tous les exemples prédictés comme appartient a cette classe , il est donné par :

$$\text{Précision} = TP/(T\text{P} * FP)$$

Recall (Rc) : le pourcentage des TP parmi tous les classe dans l'ensemble de données :

$$\text{Recall} = TP/(TP * FN)$$

F1-score (F1) : la moyenne harmonique pondérée de précision et de rappel (Recall), il est donné par :

$$\text{F1-Score} = 2 * (Pr * Rc) / (Pr+\text{Rc})$$

	Précision	Recall	F1-score	Support
Benign	97.04%	90.52%	93.66%	305
Locker	57.83%	82.75%	68.08%	83
Mediyes	98.17%	99.078%	98.62%	438
Winwebsec	99.55%	99.45%	99.50%	908
Zbot	97.66%	98.82%	98.24%	341
Zeroaccess	99.07%	94.69%	96.83%	216

Tableau 4.3 : Le rapport de notre CNN

	Précision	Recall	F1-score	Support
Benign	95.95%	91,81%	93,83%	305
Locker	72,34%	78,16%	75,89%	83
Mediyes	97,92%	97,92%	97,92%	438
Winwebsec	99,56%	99,24%	99,39%	908
Zbot	96,67%	100%	98,10%	341
Zeroaccess	98,07%	95,77%	96,90%	216

Tableau 4.4 : Le rapport de VGG16

	Précision	Recall	F1-score	Support
Benign	92.90%	93.53%	93,21%	305
Locker	73.45%	79.80%	76,49%	83
Mediyes	95.12%	95.12%	95.12%	438

Winwebsec	97.79%	98.00%	97,98%	908
Zbot	95.84%	97.28%	96,55%	341
Zeroaccess	91.90%	84.64%	88,12%	216

Tableau 4.5 : Le rapport de InceptionV3

model	Precision	Recall	F1-score
CNN 1D	91,55%	94.21%	92.48%
InceptionV3	91.33%	91.39%	91.24%
VGG16	93,41%	93.81%	93.67%

Tableau 4.6 : Comparaison des résultats des détections des malwares

4.8) Conclusion :

Nous avons implémenté trois modèles du Deep learning avec l'architecture CNN, en utilisant l'ensemble des données **Malware Detection PE-Based Analysis Using Deep Learning Algorithm Dataset** pour la détection des malwares. Le développement avait de nombreux problèmes qui nous ont fait perdre beaucoup de temps pour les résoudre.

La masse des données du data-set, le déséquilibre et t ainsi que les limites des outils matériels disponibles (processeur, mémoire). Afin de surmonter ces problèmes.Avant l'implémentation du modèle, on a fait un prétraitement pour nos données qui sont 7642 fichier exécutable,

La suppression des redondance On a obtenu des résultats très satisfaisants pour notre model une précision de détection élevée e (Accuracy = 97.12%, Loss = 0.21%) comme le but de cet model est d'assurer la sécurité donc le but ultime c'est d'avoir le moins de faux positive pour la classe Benign c'est t'a dire du moin de malware qui passent comme des benign les résultat sur l'ensemble de test sont bonne sauf pour la classe locker et c'était à cause de manque de donné pour cette classe .

Conclusion Générale

Au terme de ce travail, il nous a été possible d'avoir une vue d'ensemble sur les techniques utilisées dans le domaine de la cyber-sécurité. Nous avons démontré que l'intelligence artificielle et spécialement l'apprentissage automatique sont les méthodes les plus prometteuses pour arriver à éradiquer les malwares.

Cependant notre travail reste à parfaire sur plusieurs fronts à savoir :

-L'utilisation d'autres méthodes de deep learning en vue de comparer les méthodes entre elles

-Explorer d'autre dataset et L'affinage des procédés de prétraitement en vue de l'obtention de meilleurs ensembles de données, qui en général, sont une condition sine qua non à l'obtention de meilleurs résultats.

Bibliographie

[1]: Thomas, R. H. Core War: Creeper & Reaper
[2]: AV-TEST, “Malware Statistics & Trends Report | AV-TEST,” Av-test.org, Apr. 25, 2019. https://www.av-test.org/en/statistics/malware/ (accessed May 14, 2022).
[3] : “What are the different types of Malware? - Comtact,” Comtact, Mar. 14, 2019. https://comtact.co.uk/what-are-the-different-types-of-malware (accessed May. 15, 2022)
[4] : X. Fan, Y. Xiang, and , Defending against the propagation of active worms. Los Alamitos, California: IEEE Computer Society, 2008./
[5] : “What is a computer virus?,” SearchSecurity. https://www.techtarget.com/searchsecurity/definition/virus (accessed May. 19, 2022)
[6] : “Botnet Attacks: What Is a Botnet & How Does It Work?,” InfoSec Insights, Aug. 25, 2020. https://sectigostore.com/blog/botnet-attacks-what-is-a-botnet-how-does-it-work/ (accessed May. 18, 2022)
[7]: “What is a Trojan Virus | Trojan Horse Malware | Imperva,” Learning Center. https://www.imperva.com/learn/application-security/trojans/ (accessed May. 24, 2022)
[8]: I. Belcic, “What is Trojan Malware? The Ultimate Guide,” What is Trojan Malware? The Ultimate Guide, Nov. 19, 2021. https://www.avast.com/c-trojan (accessed May. 24, 2022)
[9] : “How Ransomware Works and How to Prevent It | ExtraHop,” www.extrahop.com. https://www.extrahop.com/company/blog/2020/ransomware-explanation-and-prevention/ (accessed May. 24, 2022)
[10] : “What is adware and how does it work?,” uk.norton.com. https://uk.norton.com/internetsecurity-malware-what-is-adware-and-how-does-it-work.html (accessed May. 25, 2022)
[11] : “What is a Trojan Virus | Trojan Horse Malware | Imperva,” Learning Center. https://www.imperva.com/learn/application-security/trojans/ (accessed May. 25, 2022)
[12] : “Phishing (hameçonnage),” www.economie.gouv.fr. https://www.economie.gouv.fr/dgccrf/Publications/Vie-pratique/Fiches-pratiques/Phishing-hameconnage#:~:text=L%27hame. (accessed May. 26, 2022)
[13]: Sosdailynews.com, 2017. https://sosdailynews.com/news.jspx?articleid=%20B39F74F035EA82E9100C308D648D6BFF (accessed May. 29, 2022)
[14]: Malwarebytes, “What is malware? Definition and how to tell if you’re infected,” Malwarebytes, 2020. https://www.malwarebytes.com/malware (accessed Jun. 03, 2022) .

[15]: Saeed, Imtithal & Selamat, Ali & Abuagoub, Ali. (2013). A Survey on Malware and Malware Detection Systems. International Journal of Computer Applications. 67. 25-31. 10.5120/11480-7108.
[16]: M. K. A, *Learning Malware Analysis : Explore the Concepts, Tools, and Techniques to Analyze and Investigate Windows Malware.* Birmingham: Packt Publishing Ltd, 2018.
[17]: T. Alsmadi and N. Alqudah, "A Survey on malware detection techniques," 2021 International Conference on Information Technology (ICIT), 2021, pp. 371-376, doi: 10.1109/ICIT52682.2021.9491765.
[18]: N. Fox, "11 Best Malware Analysis Tools and Their Features," www.varonis.com, Jan. 27, 2021. https://www.varonis.com/blog/malware-analysis-tools (accessed Jun. 05, 2022)
[19]: N. Fox, "PeStudio Overview: Setup, Tutorial and Tips," www.varonis.com, Oct. 06, 2021. https://www.varonis.com/blog/pestudio (accessed Jun. 05, 2022)
[20]: "SonarQube," *Wikipedia*, Sep. 29, 2021. https://en.wikipedia.org/wiki/SonarQube (accessed Jun. 08, 2022)
[21]: "PVS-Studio," *Wikipedia*, Mar. 04, 2022. https://en.wikipedia.org/wiki/PVS-Studio (accessed Jun. 07, 2022) .
[22]: "DeepSource - GitHub Marketplace," *GitHub*. https://github.com/marketplace/deepsource-io (accessed Jun. 10, 2022) .
[23]: "Embold - IntelliJ IDEs Plugin | Marketplace," *JetBrains Marketplace*. https://plugins.jetbrains.com/plugin/14711-embold (accessed Jun. 12, 2022)
[24]: Shalaginov, Andrii & Franke, Katrin. (2016). Automated intelligent multinomial classification of malware species using dynamic behavioural analysis. 70-77. 10.1109/PST.2016.7906939.
[25]: J. Cirelly, "10 Best Malware Analysis Tools - Updated 2022! (Paid & Free)," *Comparitech*, Mar. 16, 2022. https://www.comparitech.com/net-admin/best-malware-analysis-tools/ (accessed Jun. 14, 2022)
[26]: R. Izquierdo, "CrowdStrike Falcon Review 2022: Features, Pricing & More," *The Motley Fool*, May 18, 2022. https://www.fool.com/the-ascent/small-business/endpoint-security/crowdstrike-falcon-review (accessed Jun. 16, 2022) .
[27]: N. Fox, "Cuckoo Sandbox Overview," *www.varonis.com*, May 26, 2021. https://www.varonis.com/blog/cuckoo-sandbox (accessed Jun. 19, 2022) .
[28]: D. Kostadinov, "Reverse engineering and malware analysis tools," *Infosec Resources*, Feb. 03, 2020.

https://resources.infosecinstitute.com/topic/reverse-engineering-and-malware-analysis-tools/ (accessed Jun. 22, 2022)
[29]: A. Perlman, “The Growing Role of Machine Learning in Cybersecurity,” *SecurityRoundTable.org*, Jun. 18, 2019. https://www.securityroundtable.org/the-growing-role-of-machine-learning-in-cybersecurity/ (accessed Jun. 25, 2022)
[30]: G. Apruzzese, M. Colajanni, L. Ferretti, A. Guido and M. Marchetti, "On the effectiveness of machine and deep learning for cyber security," 2018 10th International Conference on Cyber Conflict (CyCon), 2018, pp. 371-390, doi: 10.23919/CYCON.2018.8405026 .
[31]: “Understanding AI: What Is Deep Learning? | Ironhack Blog,” *www.ironhack.com*. https://www.ironhack.com/en/data-analytics/understanding-ai-what-is-deep-learning (accessed Jun. 30, 2022)
[32]: Ravi, Vinayakumar & Alazab, Mamoun & Kp, Soman & Poornachandran, Prabaharan & Venkatraman, Sitalakshmi. (2019). Robust Intelligent Malware Detection Using Deep Learning. IEEE Access. PP. 1-1. 10.1109/ACCESS.2019.2906934.
[33]: M. Mishra, “Convolutional Neural Networks, Explained,” *Medium*, Sep. 02, 2020. https://towardsdatascience.com/convolutional-neural-networks-explained-9cc5188c4939 (accessed Jul. 03, 2022) .
[34]: “What are Convolutional Neural Networks?,” *www.ibm.com*, Oct. 20, 2020. https://www.ibm.com/cloud/learn/convolutional-neural-networks (accessed Jul. 10, 2022) .
[35]: “Papers with Code - Max Pooling Explained,” *paperswithcode.com*. https://paperswithcode.com/method/max-pooling (accessed Jul. 14, 2022) .
[36]: L. Yang, A. Ciptadi, I. Laziuk, A. Ahmadzadeh and G. Wang, "BODMAS: An Open Dataset for Learning based Temporal Analysis of PE Malware," 2021 IEEE Security and Privacy Workshops (SPW), 2021, pp. 78-84, doi: 10.1109/SPW53761.2021.00020.
[37]: M. Middleton, “Deep Learning vs. Machine Learning — What’s the Difference?,” *Flatiron School*, Feb. 08, 2021. https://flatironschool.com/blog/deep-learning-vs-machine-learning/
.**[38]**: S. Tobiyama, Y. Yamaguchi, H. Shimada, T. Ikuse and T. Yagi, "Malware Detection with Deep Neural Network Using Process Behavior," 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), 2016, pp. 577-582, doi: 10.1109/COMPSAC.2016.151.
[39]: Raff, E., Zak, R., Cox, R. *et al.* An investigation of byte n-gram features for malware classification. *J Comput Virol Hack Tech* 14, 1–20 (2018). https://doi.org/10.1007/s11416-016-0283-1 (accessed Jul. 16, 2022)

[40]: Zhou, H. (2019). Malware Detection with Neural Network Using Combined Features. In: , et al. Cyber Security. CNCERT 2018. Communications in Computer and Information Science, vol 970. Springer, Singapore. https://doi.org/10.1007/978-981-13-6621-5_8 (accessed Jul. 14, 2022) .
[41]: "Google Colab - What is Google Colab? - Tutorialspoint," *www.tutorialspoint.com*. https://www.tutorialspoint.com/google_colab/what_is_google_colab.htm (accessed Jul. 18, 2022)
[42]: Tuan, Anh Pham; Phuong, An Tran Hung; Thanh, Nguyen Vu; Van, Toan Nguyen (2018): Malware Detection PE-Based Analysis Using Deep Learning Algorithm Dataset. figshare. Dataset. https://doi.org/10.6084/m9.figshare.6635642.v1
[43]: "VirusTotal," *www.virustotal.com*. https://www.virustotal.com/gui/home/search (accessed Jun. 18, 2022)
[44]: "Malicia Project," malicia-project.com. http://malicia-project.com/ (accessed Jul. 14, 2022) (accessed Jun. 17, 2022).
[45]: C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 2818-2826, doi: 10.1109/CVPR.2016.308.
[46]: S. Liu and W. Deng, "Very deep convolutional neural network based image classification using small training sample size," 2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR), 2015, pp. 730-734, doi: 10.1109/ACPR.2015.7486599.
[47]: "ImageNet," www.image-net.org. https://www.image-net.org/
[48]: Yang, Q., Zhang, Y., Dai, W., & Pan, S. (2020). Transfer Learning. Cambridge: Cambridge University Press. doi:10.1017/9781139061773
[49]: Python Software Foundation, "What is Python? Executive Summary," Python.org, 2019. https://www.python.org/doc/essays/blurb/
[50]: J. Vaughan, "What is TensorFlow? - Definition from WhatIs.com," SearchDataManagement, Feb. 2018. https://www.techtarget.com/searchdatamanagement/definition/TensorFlow
[51]: T. rédac, "Keras : tout savoir sur l'API de Deep Learning," Formation Data Science | DataScientest.com, Jun. 18, 2021. https://datascientest.com/keras (accessed Oct. 05, 2022).
[52]: W. Urooj, "What is Python Spyder IDE and How to use it?," Edureka, May 13, 2020. https://medium.com/edureka/spyder-ide-2a91caac4e46# (accessed Oct. 05, 2022).